

Dissertation

Visually Supported Supervised Machine Learning

Christin Seifert

Graz, 2012

*Institute for Knowledge Management
Graz University of Technology*



Supervisor/First reviewer: Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt
Second reviewer: Prof. Dr. Michael Granitzer

Abstract (English)

Classification is a common task in data mining and knowledge discovery. Usually classifiers have to be generated by machine learning experts. Thus, the user who applies the classifier has no idea whether, how and why the classifier works. This lack of understanding results in a lack of trust in the algorithms. Further, excluding domain experts from the classifier construction and adaptation process does not allow to fully exploit users' domain knowledge.

In this thesis the concept of Visually Supported Supervised Learning is introduced. It is investigated whether a tighter coupling of the data mining process with the user by the means of interactive visualizations can improve construction, understanding, assessment, and adaptation of supervised learning algorithms. Different classifier-agnostic visualizations are designed and implemented and the concept of Visual Active Learning is deduced. Various experiments evaluate the suitability of these visualization with respect to assessment, understanding, creation and adaptation of classifiers.

The experiments show that, first, classifier-agnostic visualizations can help users to assess and understand arbitrary classification models in various classification tasks. Second, a specific (classifier-dependent) visualization for text classifiers can be used to assess certain aspects of the internal classification model in more detail. Third, a combination of data visualization and classifier visualization enables domain users to create classifiers from scratch. Fourth, Visual Active Learning outperforms classical active learning in classifier-agnostic settings. Fifth, automatically extracted key phrases are a fast and accurate representation for document labeling and thus allow for fast and efficient training data generation.

The results show, that the combination of classification algorithms and information visualization, Visually Supported Classification, is a reasonable approach. Tighter integration of domain users in classification applications can be beneficial for both, the users and the algorithms.

Keywords: Supervised Machine Learning, Classification, Information Visualization, Visual Active Learning, User

Abstract (German)

Klassifikation als Teilgebiet des überwachten Lernens ist ein wichtiges Gebiet des Data Minings und der Wissenserschließung. Normalerweise werden Klassifikatoren von ExpertInnen auf dem Gebiet des Maschinellen Lernens erstellt. Daraus folgt aber auch, dass die EndanwenderInnen im Allgemeinen nicht wissen, wie und warum der Klassifikator welche Entscheidungen trifft. Dieses fehlende Verständnis führt wiederum zu fehlendem Vertrauen in die Algorithmen. Außerdem ist es nicht möglich, wertvolles Domänenwissen in die Algorithmen zu integrieren, wenn man die AnwenderInnen aus dem Erstellungs- und Adaptionssprozess von Klassifikatoren ausschließt.

In dieser Arbeit wird das Konzept von visuell unterstützter Klassifikation beschrieben. Es wird untersucht, ob eine stärkere Integration von EndanwenderInnen in den Data Mining Prozess mit Hilfe von interaktiven Visualisierungen die Erstellung, das Verstehen, die Beurteilung und die Adaption von Klassifikatoren verbessern kann. Dafür werden mehrere Visualisierungen, die unabhängig vom spezifischen Klassifikator angewendet werden können, entworfen und implementiert. Weiterhin wird das Konzept des Visuellen Aktiven Lernens als Erweiterung des Aktiven Lernens im Data Mining eingeführt. In Experimenten werden diese Visualisierungen und das Visuelle Aktive Lernen hinsichtlich ihrer Verwendbarkeit für das Verstehen, die Beurteilung und die Adaption von Klassifikatoren evaluiert.

In Experimenten konnte Folgendes gezeigt werden: Erstens, die entwickelten Visualisierungen können AnwenderInnen das Verstehen und Beurteilen von Klassifikationsmodellen ermöglichen. Zweitens, eine Visualisierung für einen speziellen Textklassifikator erlaubt AnwenderInnen Zugriff auf das interne Klassifikationsmodell. Drittens, eine Kombination aus Datenvisualisierung und Klassifikatorvisualisierung ermöglicht DomänenexpertInnen, Klassifikatoren neu zu erstellen. Viertens, Visuelles Aktives Lernen liefert bessere Ergebnisse als klassisches Aktives Lernen in klassifikator-unabhängigen Fällen. Fünftens, eine Darstellung von automatisch extrahierten Schlüsselphrasen aus Texten ermöglicht ein schnelles und akkurates Annotieren von Textdokumenten und damit schnelles und akkurates Generieren von Trainingsdaten für die Textklassifikation.

Es kann geschlussfolgert werden, dass die Kombination aus Klassifikation und Visualisierung, d.h. visuell unterstützte Klassifikation, ein sinnvoller Ansatz ist. Von einer engeren Einbindung von DomänenexpertInnen in Klassifikationsanwendungen profitieren sowohl die Algorithmen, als auch die AnwenderInnen.

Acknowledgement

Lesewarnung: Dieser Teil enthält - im Gegensatz zum Rest der Arbeit - unwissenschaftliche und ironische Aussagen. Die Einordnung der Aussagen in die beiden Kategorien "ernst gemeint" und "nicht ganz ernst gemeint" wird der geneigten Leserin und dem geneigten Leser überlassen.

Mein erster Dank gilt meinen BetreuerInnen Stefanie Lindstaedt und Michael Granitzer für das hilfreiche Feedback zu dieser Arbeit. Speziell Michael Granitzer danke ich für seine bewundernswerte unendliche Geduld im Beantworten meiner Fragen und die langen Brainstorming-Sessions.

Weiterhin danke ich meine KollegInnen vom Know-Center und Institut für Wissensmanagement für die gute Zusammenarbeit, die interessanten und teilweise recht energischen Diskussionen und für die Implementierung des Bio-Mittagessens.

Insbesondere meiner Familie möchte ich danken für die mehr oder weniger dezenten, und wirklich nötigen Erinnerungen an die Arbeit, häufig in Form von beiläufigen Fragen der Art: "Wie weit bist du denn nun mit deiner Arbeit?"

Meinem Team vom Volleyballverein bin ich sehr dankbar für die wöchentlichen non-verbalen Hinweise auf die Existenz eines Lebens außerhalb der Dissertation sowie für die langanhaltenden befreienden Lachanfänge bei den Nachtrainingssitzungen.

Weiterhin danke ich Julia, Katrin und Veronika (in alphabetischer Reihenfolge) für ... nun, das erzähle ich ihnen persönlich.

Außerdem danke ich meinem Hund, den ich nicht besitze, für die viele Zeit, die ich durch nicht durchgeführte Spaziergänge für das Schreiben dieser Arbeit verwenden konnte.

Christin Seifert
Graz, 2012

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz,

Place, Date

Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am

Ort, Datum

Unterschrift

Contents

Abstract (English)	iii
Abstract (German)	v
Acknowledgement	vii
Contents	xi
List of Figures	xv
List of Tables	xvii
List of Algorithms	xix
1 Introduction	1
1.1 Research Question	3
1.2 Methodology	4
1.3 Focus of the Thesis	5
1.3.1 Demarcation	7
1.4 Contributions	8
1.5 Publications	9
1.6 Terminology	11
1.7 Outline	11
2 Foundations	15
2.1 Information Visualization, Visual Analytics, Visual Data Mining	15
2.2 Supervised Learning – Classification	17
2.2.1 Types of Classification Problems	18
2.2.2 Trivial Classifiers	19
2.2.3 K-Nearest Neighbor Classifier (KNN)	20
2.2.4 Naive Bayes Classifier (NB)	21
2.2.5 Support Vector Machines (SVMs)	22
2.2.6 Class-Feature-Centroid Classifier (CFC)	23
2.2.7 A-Posteriori Probabilities	24
2.2.8 Summary	24
2.3 Text Classification	25

Contents

2.4	Performance Evaluation for Classifiers	27
2.4.1	Evaluation Methods	27
2.4.2	The Contingency Table	28
2.4.3	Binary Classification	29
2.4.4	Generalization to Multi-Class Problems	30
2.4.5	Summary	33
2.5	Active Learning	33
2.6	Voronoi Diagrams	34
2.7	Summary	35
3	Theory	37
3.1	Towards Visualizations for Arbitrary Classifiers	41
3.1.1	Common Properties of Classifiers	41
3.1.2	Properties of a Classifier-Agnostic Visualization	43
3.1.3	State-of-the-Art	44
3.1.4	Summary	47
3.2	Towards Summarized, Visual Representations of Text	51
3.2.1	State-of-the-Art	52
3.2.2	Summary	52
3.3	Towards Visualizations for Text Classification	54
3.3.1	Tag Layout	55
3.3.2	Summary	56
3.4	Towards Visual Active Learning	58
3.4.1	Active Learning	58
3.4.2	Concept	59
3.4.3	Comparing Active Learning and Visual Active Learning	60
3.4.4	Requirements for a Visualization for Visual Active Learning	60
3.4.5	Summary	61
4	Implementation	63
4.1	Class Radial Visualization	65
4.1.1	Method	65
4.1.2	Special Views	68
4.1.3	Embedding in an Application	71
4.1.4	Parameters	72
4.1.5	Feedback Example	72
4.1.6	Properties	72
4.1.7	Supported Tasks	74
4.1.8	Suitability for Visual Active Learning	75
4.1.9	Comparison to Related Work	76
4.1.10	Summary	76
4.2	Confusion Maps	78
4.2.1	Method	78
4.2.2	Parameters	78

4.2.3	Examples	79
4.2.4	Properties	79
4.2.5	Summary	81
4.3	Tag Clouds	82
4.3.1	Method	82
4.3.2	Parameters	87
4.3.3	Examples	88
4.3.4	Properties	90
4.3.5	Summary	90
4.4	Voronoi Word Clouds	91
4.4.1	Method	91
4.4.2	Parameters	92
4.4.3	Examples	92
4.4.4	Properties	92
4.4.5	Summary	93
4.5	Summary	94
5	Experiments	95
5.1	Data Sets	98
5.1.1	Iris Flower Data Set	98
5.1.2	COIL-20 Image Data Set	98
5.1.3	Reuters-21578 Text Corpus	99
5.1.4	APA Blog and News Corpus	100
5.1.5	20 Newsgroup Corpus	101
5.2	Experiment Set 1: Quality Assessment of Classifiers	103
5.2.1	Quality Assessment for Image Classification	103
5.2.2	Quality Assessment for Cross-Domain Text Classification	107
5.3	Experiment Set 2: Visual Active Learning	113
5.3.1	Single User Experiment	114
5.3.2	User Simulations	120
5.4	Experiment 3: Document Representation for Efficient Labeling	131
5.4.1	Procedure	131
5.4.2	User Evaluation	133
5.4.3	Results	138
5.4.4	Discussion	144
5.4.5	Summary	146
5.5	Experiment 4: Visual Hypothesis Generation	147
5.5.1	Procedure	147
5.5.2	Interaction Methods	149
5.5.3	Results	150
5.5.4	Discussion	152
5.5.5	Summary	153
5.6	Experiment 5: Visualizing Text Classification Models	155
5.6.1	Procedure	155

Contents

5.6.2	Results	156
5.6.3	Discussion	157
5.6.4	Summary	157
5.7	Summarizing the Experiments	158
6	Conclusion and Future Work	159
6.1	Assessment	160
6.2	Directions for Future Work	162
	List of Abbreviations	163
	Bibliography	165

List of Figures

1.1	Methodology of the thesis	6
1.2	Scope of the thesis	6
1.3	Focus of the thesis: Knowledge Discovery view	7
1.4	Focus of the thesis: Information Visualization view	7
1.5	Outline of the thesis	12
2.1	Overview of the Visual Analytics Process	16
2.2	Text classification pipeline	25
2.3	The active learning scenario	33
2.4	Example Voronoi Diagram	35
3.1	Views of a classifier	41
3.2	Example classifier hypothesis	42
3.3	Example classifier-agnostic, visualizations for two classes.	44
3.4	Example classifier-agnostic visualizations for multiple classes	45
3.5	Overview of the labeling process for text classification	51
3.6	Examples of visual text representations	53
3.7	Example tag layouts	57
3.8	The Visual Active Learning scenario	59
4.1	Histograms of different examples	67
4.2	Layout for the examples of figure 4.1	67
4.3	Magic lens view of the Class Radial Visualization	69
4.4	Misclassification view of the Class Radial Visualization	70
4.5	Screenshot of the application integrating the Class Radial Visualization	71
4.6	The feedback process with the Class Radial Visualization	73
4.7	Examples visualizations revealing different phenomena	75
4.8	Example Confusion Maps	79
4.9	Comparison of Confusion Matrices and Confusion Maps	80
4.10	Overview of the tag layout algorithm	83
4.11	Methods to decrease the size of tags' bounding boxes	84
4.12	The rectangle layout process	86
4.13	Example tag layouts in arbitrary convex shapes	89
4.14	Example Voronoi Word Cloud	93
5.1	Objects of the COIL-20 image data set	99
5.2	Classifier visualization, COIL-20, PCA features, KNN and NB	104

List of Figures

5.3	Classifier visualization, COIL-20, LDA features, KNN and NB	104
5.4	Comparing classifier visualizations for cross-domain classification task . .	110
5.5	Comparing Confusion Maps for cross-domain classification task.	111
5.6	Visualizing misclassification for the cross-domain classification task . . .	112
5.7	Screenshots classifier visualizations, random selection vs user selection . .	118
5.8	Accuracy plots. Comparing random selection and user selection	119
5.9	Overview of experimental procedure for user simulations	122
5.10	Models of user selection strategies	124
5.11	Performance plots: Comparing selection strategies	125
5.12	Performance plots: Comparing labeling strategies	127
5.13	Overview of the evaluation methodology	131
5.14	Overview of the evaluation procedure	134
5.15	Box plot of the word distribution	136
5.16	Screenshot: full-text condition	137
5.17	Screenshot: key sentences condition	137
5.18	Screenshot: key phrases condition	138
5.19	Box plots for task completion time and number of correct labels	139
5.20	Histograms of the number of correct labels	140
5.21	Histograms for the task completion times	140
5.22	Infographics comparing perceived and measured performance	145
5.23	Process overview for creating classifiers using data visualization	148
5.24	Screenshot of the text classification application for creating classifiers . . .	151
5.25	Voronoi Word Clouds for CFC classifier on 20NEWS data set	156

List of Tables

2.1	Dimensions of classification problems	19
2.2	Contingency table for binary classification problems	28
2.3	General class confusion matrix	31
2.4	Example for ground truth and predictions	32
2.5	Example class contingency tables	32
2.6	Class confusion matrix for example in table 2.4	32
3.1	Aspects of classifiers for understanding and feedback	38
3.2	Overview of existing classifier visualizations	48
3.2	Overview of existing classifier visualizations (contd.)	49
3.2	Overview of existing classifier visualizations (contd.)	50
3.3	Comparison of Active Learning and Visual Active Learning.	60
4.1	Overview of all developed visualization	63
4.2	Comparing Visualix and Class Radial Visualization	77
4.3	Overview quality measures for tag layout	88
5.1	Overview of all experiments	97
5.2	Overview of the Iris Flower data set	98
5.3	Overview of the splits of the COIL-20 data set	99
5.4	Overview of the R8 subset of the Reuters-21578 data set	100
5.5	Overview of the APA News Topic Dataset	101
5.6	Overview of the APA Blog Topic Dataset	101
5.7	Overview of the 20 Newsgroup data set	102
5.8	Accuracy, training and classification time for scenario <i>NewsBlog</i>	109
5.9	Accuracy random selection and user selection, Iris and MLP	115
5.10	Accuracy random selection and user selection, COIL-20 and KNN	116
5.11	Accuracy random selection and user selection, COIL-20 and SVM	116
5.12	Accuracy random selection and user selection, REU-R8 and SVM	116
5.13	Accuracy random selection and user selection, APA and CFC	117
5.14	Overview of the results comparing random selection and user selection	118
5.15	Overview of compared data set and classifier combinations	123
5.16	Overview of compared selection strategies	123
5.17	Effect of labeling only misclassified examples	128
5.18	Overview of labeling time and number of correct labels	139
5.19	Comparing classifier accuracy with original labels and user labels	141

List of Tables

5.20 Comparing original labels and user labels	142
5.21 Overview of questionnaire answers	143
5.22 Interaction Methods in Information Landscape and classifier window . . .	150

List of Algorithms

1	Layout algorithm for the Class Radial Visualization	66
2	Algorithm pseudo-code for the tag layout algorithm Trunc-Shift-Scale . . .	87
3	Overall procedure of the experiments on Visual Active Learning	114

“What we’ve got here is a failure to communicate.”

(Captain (Strother Martin) in Cool Hand Luke)

1 Introduction

Classification is a common task in data mining and knowledge discovery. Application include spam-filtering of emails, categorization of web pages, gene-sequence classification and object recognition.

The generation of a good classifier is a hard task, usually performed by data mining experts. Features have to be engineered, training data has to be generated and an appropriate classification algorithm has to be selected, parametrized and trained. Unfortunately, as stated by the no-free-lunch theorems [Wol96b], there is no such thing as the best classification algorithm. Empirical studies may give a hint which algorithm to use for a given problem. Experts in data mining know which algorithm surely will not work in a given setting. But at the end, it comes down to testing multiple classifiers and evaluating them on the task at hand to find the best performing one.

Given that the classifiers are created by data mining experts, the user who applies the classifier has usually no idea whether, how and why the classifier works. Sometimes, the performance of classifiers is accessible to users only, if they detect obvious misbehavior while applying the classifier. Performance of a classifier means its efficiency on real-world data in real-world applications, i.e. how often does the classifier make which mistakes.¹ But still, the classifier is a black-box for users, many questions about its behavior remain unanswered. Thus, generally, users of classifiers do not understand why the classifier is doing what and how well it is performing. Further, it is questionable why users then should trust classifiers. Understanding and trust of data mining models has been identified as desirable property of the models [TBD⁺01]. This is the reason why less powerful, but easy to communicate classification models such as decision trees are in some applications preferred to very powerful classification models, like artificial neural networks and support vector machines [Koh00].

There is another problem with the current practice of users applying black-box classifiers generated by data mining experts. Besides the lack of understanding and trust, this approach does not exploit the potential of the user – specifically the domain knowledge. Domain knowledge can not always be encoded in machine-readable form and integrated

¹Performance in terms of computational complexity is equally important but not the focus of this thesis.

1 Introduction

into the algorithms. Further, domain knowledge may become explicit only in the process of working with the data and the algorithm. Thus, excluding domain users from the classifier construction and adaptation process does not allow to fully exploit users' domain knowledge.

As argued above, current practice of classifier generation leads to (i) lack of understanding and trust for end users and (ii) little or no exploitation of domain knowledge for classifiers.

These problems have already been identified in the literature, most prominently by Ben Shneiderman [Shn02]. The general approach to solving this problem is to tighter couple the automatic approaches and the user using information visualizations. Appropriate interactive visualizations can be used to (i) convey details about the data mining process to the user, and subsequently generate trust and understanding [KT03], and (ii) integrate background knowledge of the user into the algorithm [WEH⁺01].

Existing approaches to combine information visualization and classifiers are either tailored towards specific classifiers (e.g., [CCWH08, MK08a]) or otherwise restricted. Other restrictions are for instance the limitation to binary classification problems (e.g., [Rd00, FH03]) or the non-interactivity of the visualizations, such that user feedback to the model is not possible (e.g. [KLM⁺00, DA08]). It is important that the visualization is independent of the classifier, i.e. the actual classification model should be oblivious to the user. This independence of classifiers is desired in order to require the user to learn only one visualization and in order to compare classifiers by the means of the visualization.

In this thesis it is investigated whether a tighter coupling of the data mining process with the user by the means of interactive visualizations can improve construction, understanding, assessment, and adaptation of classifiers. A detailed discussion on what construction, understanding, assessment, and adaptation of classifiers means can be found in the next section where the research question of the thesis is defined. Thereby the focus lies on using classifier-agnostic interactive visualizations.

Thus, in this thesis, two classifier-agnostic visualizations are designed and implemented. Further, the concept of Visual Active learning is derived, as an extension of classical active learning. Various experiments evaluate the suitability of these visualizations to improve assessment, understanding, creation and adaptation of classifiers. Furthermore, in the experiments Visual Active Learning is compared to classical active learning to answer the question whether this concept – which gives the user more power in the process – can lead to improved classifiers.

During the thesis work and the experiments another problem became obvious, which is not directly related to interactive visualizations of classifier models. Especially for text classification problems we found that the time for generating the training data (which relates to classifier construction) heavily depends on the time the user requires to comprehend the text. Thus, the overall time required for constructing the classifier depends on the time used for text comprehension. We exploit different forms of text

representations to investigate whether this time can be reduced by altering the way the text is presented to the user. These presentations are visualizations of certain aspects of texts, i.e., they are visualizations of the data required for classification, not visualizations of the classification models. In user studies these developed data visualizations are evaluated for their suitability for faster training data generation.

1.1 Research Question

The aim of this thesis is to bridge the gap between classification algorithms and users. The chosen means to achieve this goal are interactive visualizations. More specifically, the research question this thesis aims to answer is:

RQ “Can interactive visualization improve construction, understanding, assessment, and adaptation of supervised machine learning algorithms?”

In the following we will describe in detail what “construction”, “understanding”, “assessment” and “adaptation” of supervised machine learning algorithms mean. In this thesis, the term classifier is used to refer to a supervised machine learning algorithm.²

Construction of Classifiers: Usually, classifiers are constructed by (i) selecting an appropriate algorithm, (ii) defining the algorithm’s parameters, if needed, and (iii) train the classifier by providing a training data set.³ Algorithm selection and parameter settings are performed by machine learning experts, usually in an iterative way by evaluating different combinations of algorithm and parameters. The creation of the training data set happens independently by domain experts. What shall be investigated in this thesis is whether the steps can be tighter coupled and domain users can be enabled to construct their classifiers themselves by the use of interactive visualizations. This means, they should be enabled to take an algorithm, set the parameters (or take default parameters), generate the training data and provide the training data to the classifier in order to construct a trained classification model.

Assessment of Classifiers: Once a classifier has been trained it is desirable to assess its quality and behavior. Usually, this is again a task for machine learning experts. Using different evaluation measures machine learning experts can say whether the classifier performs good or bad and, if it performs bad, they may be able to point out reasons for it after investigating the model beyond simple evaluation measures. Domain users usually do not have the means to assess classifiers at all. In terms of this thesis it will be investigated whether interactive visualization (i) can aid machine learning experts, and (ii) can be a means for domain experts in order to assess classifiers. Typical results of assessing a classification model include: How many training samples were used to build

²Classifiers and classification are known in many research fields such as biology. In this thesis we use the term classifier from the field of machine learning.

³Feature engineering is equally important, but in the scope of this thesis it is assumed that features are already engineered.

1 Introduction

the current classifier? Were there enough? How does the classifier label the test examples? How well does the classifier perform? Are there any conspicuousnesses regarding the classes or the samples? What was the decision for a sample based on? Additionally, observing a classification procedure could enable the user to answer questions like: How many training samples are necessary to get a stable classifier? Are there any “problem” samples for which the classifier constantly changes its decision?

Understanding of Classifiers: The assessment of the classifier and being able to answer the questions described above may eventually lead to some kind of understanding of the classification model. Users may be able to draw conclusions about the classification model and be able to argue why the classifier is doing what. Further, they may be able to understand when and when not to use the classifier and to recognize when it performs wrongly. Understanding a classification model is a prerequisite to communicate classifiers. Communicating classifiers means to express its behavior, performance and features in natural language understandable by lay persons. The importance of understanding models should not be underestimated. Understanding is crucial to generate trust in the models. And why should one use a model that one does not trust?

Or, as Caragea et al. put it: “Although the predictive accuracy is an important measure of the quality of a classification model, for many data mining tasks understanding the model is as important as the accuracy of the model itself.” [CCWH08]

Adaptation of Classifiers: The adaptation of classifiers is related to the construction of classifiers. Adaptation implies to correct wrong decisions or provide additional information to update the classifier’s internal models. Clearly, assessment and understanding are the basis for adaptation. A user can only corrects decisions once she assessed (and optionally) understood them. Adaptation aims at enhancing already trained models.

1.2 Methodology

In this section the methodology used for this thesis work is described. An overview of the pursued steps can be seen in figure 1.1. The general problem that the thesis tries to tackle is bridging the gap between classification algorithms and the user. Using interactive visualization seems a reasonable approach (①) and has been applied already in this context. State-of-the art research leads to research hypotheses (②). These hypotheses were at first abstract hypotheses of the form “Can interactive visualizations ..?”

In order to formulate more concrete hypotheses, which can be tested in practice, the abstract ”interactive visualizations“ was substituted with concrete visualizations. Therefore, based on the state-of-the art research, different visualizations and one interaction concept were developed as modules (③). These modules are

M1 Class Radial Visualization

M2 Confusion Maps visualization

M3 Tag Layout

M4 Voronoi Word Cloud visualization

M5 The concept of Visual Active Learning

To test the concrete hypotheses multiple experiments were designed (④) to evaluate the construction, understanding, and adaptation part of the research questions. More specifically, experiments to answer the following questions were performed⁴:

Exp1 In which way can visualizations, more specifically the Class Radial Visualization and the Confusion Maps help experts to understand arbitrary classification models?

Exp2 Can user feedback on classification models through interactive visualizations, more specifically an interactive version of the Class Radial Visualization, be used to improve classification models? Is there a benefit over automatic methods?

Exp4 Can pure data visualizations be used to allow domain experts to generate their own classifiers?

Exp5 In which way can a model of a specific text classifier, namely the [Class-Feature-Centroid \(CFC\)](#) classifier be visualized and made accessible to users?

From the results and observations of the experiments, especially Exp2 and Exp4, a crucial bottleneck for classifier adaptation and construction was identified: Using the interactive visualization was relatively quickly accomplished by users. In terms of interaction methods users were able to instantly provide feedback to the classifier. However, for text classification tasks the bottleneck was the time needed by users to actually understand and categorize the content of a text document (⑤). This insight resulted in a new hypothesis (⑥) and the design of a new experiment to test this hypothesis (⑦):

Exp3 What are good representations of the data to classify, more specifically of text documents, to speed-up the manual labeling process?

1.3 Focus of the Thesis

This section defines the focus of the thesis and names relevant research fields. As visualized in figure 1.2 this thesis covers three main research areas, Information Visualization, (supervised) Machine Learning, and Human Computer Interaction. The broad scope of the thesis is the intersection of these research fields *Information Visualization*, *Machine Learning* and *Human Computer Interaction*.

⁴The numbering of the experiments corresponds to the sequence in which they are described in the experiment section of this thesis

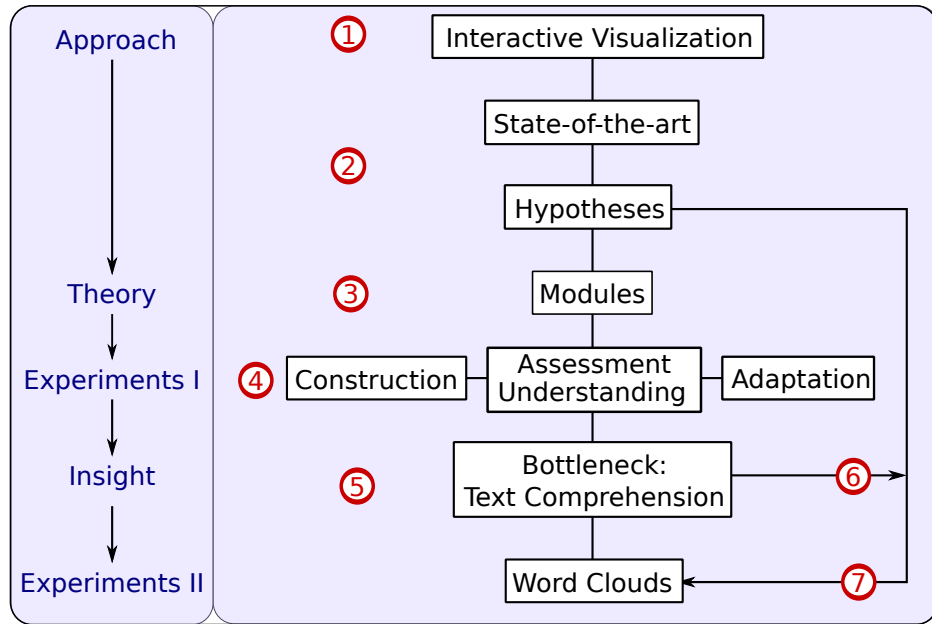


Figure 1.1: Overview of the methodology of the thesis. ①The approach to answering the research question is investigating interactive visualizations. ②Reviewing state-of-the-art literature leads to research hypotheses. ③To test the hypotheses suitable modules (i.e. visualisation components, interaction concepts and feedback concepts) are developed and implemented. ④In the first series of experiments the hypotheses concerning classifier assessment, understanding, adaptation and construction are verified using the modules. ⑤The first series of experiments lead to the insight that a crucial bottleneck for classifier adaptation and construction is human text comprehension, which ⑥lead to an new hypothesis and a ⑦second experiment to test the new hypothesis.

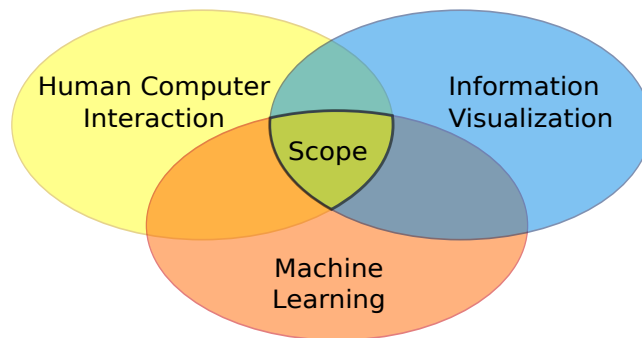


Figure 1.2: The scope of the thesis visualized in a Venn diagram. The scope is the intersection of three research fields.

1.3.1 Demarcation

In terms of the Knowledge Discovery pipeline [FPSS96] depicted in figure 1.3 the focus of this thesis is inside the blue boundary (large rounded rectangle), covering the *Data Mining* and the *Evaluation and Interpretation* step. In terms of the extended Information Visualization pipeline [MK08a] depicted in figure 1.4 the thesis focuses on the feedback loop inside the magenta border (large dashed rectangle). More specifically the thesis covers the steps *Interaction in Visualization*, *Classifier Update* and *Classifier Model*.

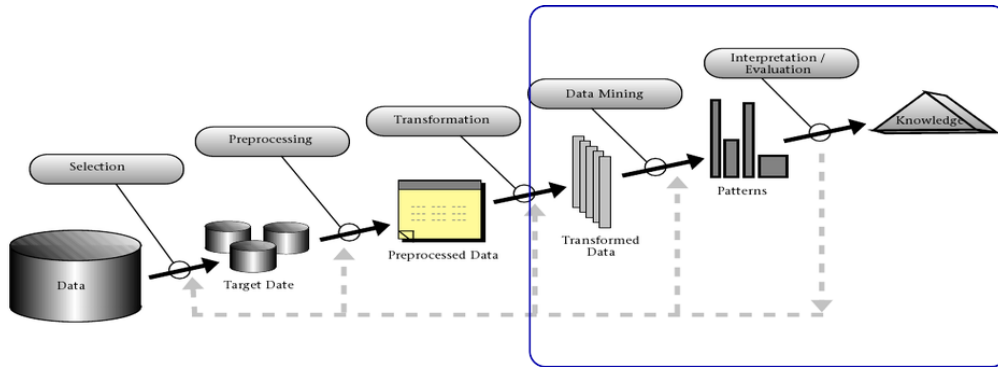


Figure 1.3: Focus of the thesis (blue boundary) in terms of the Knowledge Discovery Pipeline [FPSS96].

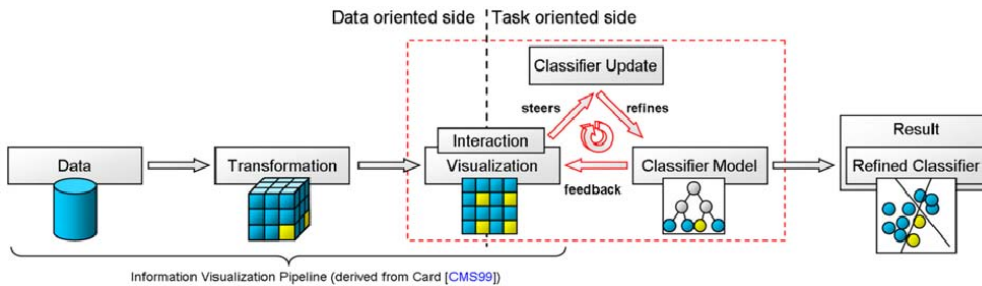


Figure 1.4: Focus of the thesis in terms of the extended Information Visualization pipeline [MK08a].

In detail the demarcation of this thesis is as follows:

- This thesis focuses on single-label, multi-class classification problems, multi-label classification will only be briefly touched.
- We investigate supervised learning, more specifically classification; semi-supervised learning is not the focus of this thesis.
- The focus application area is text classification. Some experiments use other data sets as well to show the general applicability of the visualizations and concepts.
- For this thesis we assume that preprocessing (e.g. feature engineering) is already finished and we start with example-feature matrices.

1 Introduction

- In terms of Information Visualization the focus lies on classifier visualization that are independent of the actual classifier used. The reason is threefold: First, only a small number of visualizations have to be developed which reduces the development time. Second, the user only needs to learn to interpret only few visualizations and does not need to adapt to new visualizations. Third, the usage of standardized visualizations allows to visually compare different classifiers.

1.4 Contributions

The contributions of this thesis are the following:

Contribution 1: Interactive Classifier-Agnostic Visualization: Common properties of classifiers are identified. Desired properties of interactive visualizations for classification models are derived from the tasks assessment, understanding, construction and adaptation of classifiers. Combining the desired properties with common properties of classifiers leads to the design and implementation of two classifier-agnostic visualizations. The suitability of these visualizations for assessment, understanding, construction and adaptation is confirmed in various experiments.

Contribution 2: Concept of Visual Active Learning: The concept of Visual Active Learning is developed as an extension of classical active learning. Experiments on various classifier-data set combinations using the developed classifier-agnostic visualizations prove the concept. Further, the experiments show that classical active learning is – at least for the tested combinations of classifiers and data sets – outperformed by Visual Active Learning. This finding points toward the beneficialness of integrating the user in data mining processes.

Contribution 3: Classifier-Dependent Visualization for Text Classification: For the special case of text classification a classifier-dependent visualization was developed to visually access the feature level. This visualization shows which features contribute to the classification model in which way and how the trained classes relate to each other in terms of the features used by the classifier.

Contribution 4 - Tag layout Algorithm for Arbitrary Convex Shapes: Supporting for the visualization in Contribution 3 a new tag layout algorithm was developed. This allows the space-filling layout of tags or words inside arbitrary convex shapes.

Contribution 5: Evaluation of text representations for faster labeling: For the purpose of minimizing the time required to generate training data for text classification alternative text representation forms were investigated. More specifically, only the key sentences or the key words of the texts were presented to the users. The latter one was represented as a word cloud using the layout algorithm of Contribution 4. In a user evaluation these representations were compared to the commonly used full-text. It was shown that the key word representation allows users to label training data accurately and fast.

1.5 Publications

This section summarizes my own and joint publications and outlines how they relate to this thesis.

The idea of the *classifier visualization* was first published in 2009 at the Information Visualization conference and later extended with the idea of feedback as a poster at the EuroVis conference. Both publications are a shorter version of Section 4.1 on page 65.

Christin Seifert and Elisabeth Lex. A novel visualization approach for data-mining-related classification. In *Proc. of the International Conference on Information Visualisation (IV)*, pages 490–495. Wiley, July 2009. (see [SL09a])

Christin Seifert and Elisabeth Lex. A visualization to investigate and give feedback to classifiers. In *Proceedings European Conference on Visualization (EuroVis)*, Jun 2009. poster. (see [SL09b])

In the following joint work, *classifiers were evaluated* for a cross-domain text classification task. In the context of this work, the Class Radial Visualization was supportively used to assess the quality of different classifiers. My contribution was the application of the Class Radial Visualization to the data sets and guiding the interpretation. The findings are summarized in the experiments chapter, in section 5.2.2 on page 107. Further, in the third publication, the *class confusion map visualization* was introduced.

Elisabeth Lex, **Christin Seifert**, Michael Granitzer, and Andreas Juffinger. Cross-domain classification: Trade-off between complexity and accuracy. In *Proceedings of the 4th International Conference for Internet Technology and Secured Transactions (ICITST)*, 2009. (see [LSGJ09a])

Elisabeth Lex, **Christin Seifert**, Michael Granitzer, and Andreas Juffinger. Automated blog classification: A cross-domain approach. In *Proc. of IADIS International Conference WWW/Internet*, 2009. (see [LSGJ09b])

Elisabeth Lex, **Christin Seifert**, Michael Granitzer, and Andreas Juffinger. Efficient cross-domain classification of weblogs. *International Journal of Computational Intelligence Research*, 1(1):7382, 2010. (see [LSGJ10])

An application allowing users to *construct classifiers* from scratch on their data has been proposed in the following publication. In this publication the Information Landscape from [SKM⁺09] was combined with a classification interface and the classifier visualization. Application and results are described in section 5.5 on page 147.

Christin Seifert, Vedran Sabol, and Michael Granitzer. Classifier hypothesis generation using visual analysis methods. In Filip Zavoral, Jakub Yaghob, Pit Pichappan, and Eyas El-Qawasmeh, editors, *Networked Digital Technologies, volume 87 of Communications in Computer and Information Science*, pages 98–111. Springer, 2010. (see [SSG10])

1 Introduction

Based on the classifier visualization the concept of *user-based active learning*⁵ was proposed in the workshop on Visual Analytics and Knowledge Discovery at the International Conference in Data Mining (ICDM). Section 3.4 of this thesis describes the concept in more detail and the experiments are described in section 5.3 on page 113.

Christin Seifert and Michael Granitzer. User-based active learning. In Wei Fan, Wynne Hsu, Geoffrey I. Webb, Bing Liu, Chengqi Zhang, Dimitrios Gunopulos, and Xindong Wu, editors, *Proceedings of 10th International Conference on Data Mining Workshops (ICDM2010)*, pages 418–425, Sydney, Australia, Dec 2010. (see [SG10])

The visualization Voronoi Word Cloud was applied to the CFC classifier, *visualizing the model* of this specific classifier. The Voronoi Word Cloud was presented as a poster. It builds upon previous work on tag layout in arbitrary polygons. The tag layout algorithm is described in detail in section 4.3 on page 82; the general idea of the Voronoi Word Cloud can be found in section 4.4 on page 91 and its application to the special classifier in section 5.6 on page 155.

Christin Seifert, Barbara Kump, Wolfgang Kienreich, Gisela Granitzer, and Michael Granitzer. On the beauty and usability of tag clouds. In *Proceedings of the 12th International Conference on Information Visualisation (IV)*, pages 17–25, Los Alamitos, CA, USA, July 2008. IEEE Computer Society. (see [SKK+08])

Christin Seifert, Wolfgang Kienreich, and Michael Granitzer. Visualizing text classification models with Voronoi Word Clouds. In *Proceedings 15th International Conference Information Visualisation (IV)*, 2011. poster. (see [SKG11])

Further, the experiment investigating effective text representations for *fast training data generation* has been presented recently at the Discovery Science conference.

Christin Seifert, Eva Ulbrich, and Michael Granitzer. Word clouds for efficient document labeling. In *The Fourteenth International Conference on Discovery Science*, October 2011. (see [SUG11])

Finally, a book chapter about Visual Analytics for text has been accepted and will appear in 2012 in the book Large Scale Data Analytics. This chapter covers the whole Knowledge Discovery Pipeline [FPSS96], whereas the focus of this thesis is a part of the pipeline (see section 1.3).

Christin Seifert, Vedran Sabol, Wolfgang Kienreich, Elisabeth Lex, and Michael Granitzer. Gkoulalas-Divanis, A. and Labbi, A. (Eds.) Large Scale Data Analytics Visual Analysis and Knowledge Discovery for Text *Springer*, toAppear (see [SSK+ar])

⁵User-based active learning is synonymously used to Visual Active Learning

1.6 Terminology

Depending on the historical background of the data mining algorithm different terms are used in the literature. Here we describe the terminology used throughout this thesis.

First of all, in this thesis the phrase *supervised machine learning* is usually replaced by the term *classification* – which is a synonym in the field of machine learning. The type of algorithm that is the focus of this thesis, a *supervised machine learning algorithm*, is referred to as *classification algorithm* throughout this thesis. We use the term *classifier* for both, the learning component and the classification component. A trained classifier has learned a model of the training data and is therefore referred to as *classification model*.⁶

We will refer to the object that is to be classified as *(data) item* or *example*. Data items used for creating (training) a classifier are referred to as *training items* or *training examples*. All training items form the *training set*. A specific feature of the training items is that they have a *class label* assigned.⁷ This class label describes the category the corresponding object in the real world belongs to. In this thesis, data items used for the evaluation of a classifier are called *evaluation items* and form the *evaluation set*.⁸ Evaluation items also have a label assigned. In some training scenarios, a third set of items, the *test set*, is used. *Test items* usually do not have a label. To emphasize this fact, they are sometimes also called *unlabeled data items*. Data items are represented by *features* to make them processable by machine learning algorithms.⁹

Furthermore, the term *visualization* is used synonymously to *information visualization*.

1.7 Outline

This thesis consists of 6 chapters. Figure 1.5 gives an overview of the chapters, which are also described in the following:

Chapter 1 - Introduction: Introduces and motivates the work of this thesis. Defines the research question. Describes scope of the thesis as well as goal and non-goals. States the contributions. Introduces terminology used in this work. →page 1

Chapter 2 – Foundations: Describes the foundations necessary to understand this thesis. Can be skipped by readers familiar with the following topics: Supervised learning (classification), k-Nearest Neighbor (KNN) classifier, Support Vector Machine (SVM) classifier, Naive Bayes (NB) classifier, CFC classifier, text classification, active learning and Voronoi diagrams. →page 15

⁶An alternative term is classification hypothesis.

⁷The neural network community uses the term target for labels of training items.

⁸In other contexts, the evaluation or validation set is used for parameter estimation of algorithms.

⁹On other contexts features are also referred to as attributes.

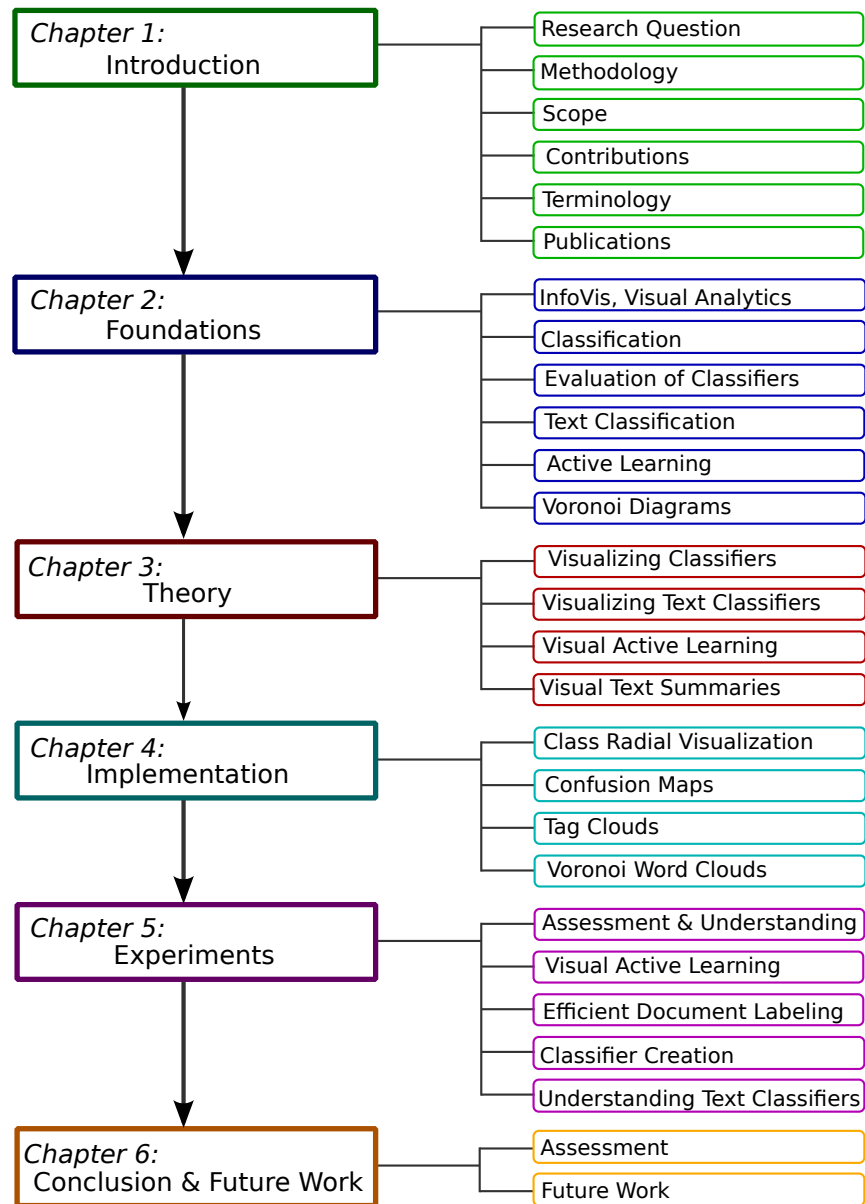


Figure 1.5: Overview of the outline of this thesis

Chapter 3 – Theory: Develops theory of visually supported supervised learning. Identifies requirements for visualizations supporting visual supervised learning. Identifies common properties of classifiers. Reviews state-of-the art. Sketches visualizations. Develops the concept of visual active learning. →page 37

Chapter 4 – Implementation: Describes implementation details for the visualizations sketched in chapter 3. Describes the developed Class Radial Visualization, Confusion Maps, a specific tag layout algorithm and Voronoi Word Clouds. →page 63

Chapter 5 – Experiments: Describes five experiments performed to answer the research question using the visualizations developed in chapter 3 and implemented in chapter 4. More specifically, the experiments cover assessment and understanding of classifiers (experiment in section 5.2 and 5.6), Visual Active Learning (experiment in section 5.3), efficient training data generation for text classification (experiment in section 5.4), and creation of classifiers (experiment in section 5.5). →page 95

Chapter 6 – Conclusion and Future Work: Summarizes the work of the thesis with respect to the research question and goals. Self-assessment of the achieved results. Gives directions for future work. →page 159

“An investment in knowledge pays the best interest.”

(Benjamin Franklin)

2 Foundations

This chapter describes the foundations necessary to understand the concepts of the theory chapter and the experiments. *Those familiar with the topics, can omit this chapter.*

The chapter is structured as follows: The terms information visualization, visual analytics and visual data mining are defined and explained in section 2.1 on page 15. Classification as a concept of supervised machine learning is introduced in section 2.2 on page 17. This section includes the definition of types of classification problems, the concept of trivial classifiers and describes in detail the four different classification algorithms that are used in this thesis. Specialties for text classification, mostly the feature generation part are covered in section 2.3 on page 25. Evaluation methodology and measures for classifiers are described in section 2.4 on page 27. The concept of active learning is explained in section 2.5 on page 33. Voronoi diagrams are briefly explained in section 2.6.

2.1 Information Visualization, Visual Analytics, Visual Data Mining

As described in section 1.3 this thesis roughly covers the research fields Information Visualization, classification as a subfield of Machine Learning by taking the user into account. How these research fields relate to Visual Analytics and Visual Data Mining will be described in this section.

Defining Visualization can be very simple or very hard. The simplest definition of Visualization is: ‘If you can see it, it’s a visualization‘ (Pat Hanrahan, Keynote at the European Conference of Visualization, 2009). According to this definition, a car, a cloud and a TreeMap [Shn92] all are visualizations.

The InfoVis wiki ¹ defines **Visualization** as ”A graphical representation of data or concepts, which is either an internal construct of the mind or an external artifact supporting

¹<http://www.infovis-wiki.net>

decision making.“ Note, that according to this definition visualizations not necessarily make use of computers (as opposed to many other definitions which explicitly define the use of computer hardware).

Visualization can be further subdivide into Scientific Visualization, Information Visualization and, more recent Knowledge Visualization [EB04]. Basically, the difference between these three is the kind of data that is represented. **Scientific visualizations** represent scientific data, like for instance measurements, in the original data space. Examples of scientific visualizations are time series of measurements of size of a glacier or the speed of a vehicle. Scientific visualizations reflect the data precisely. **Information visualizations** [CMS99, War04, Spe06] represent more abstract data. These visualizations need not necessarily be precise, they aim at conveying latent information in the data. Examples of information visualizations are the TreeMap [Shm92] – visualizing directory structure and content, and PhraseNets [vHWW09] – visualizing document structure. Keim et al.[KMSZ06] defines Information Visualization as follows: ”Information visualization (InfoVis) is the communication of abstract data through the use of interactive visual interfaces.“ **Knowledge visualizations** visualize even more abstract concepts – namely knowledge. The main purpose of knowledge visualizations is communication. An example is the TubeMap visualization for project management [BM05].

Visual Analytics [TC05] can be seen as an extension of Information Visualization by including the human in the visualization process. Thomas and Cook define Visual Analytics as follows: ”Visual analytics is the science of analytical reasoning facilitated by interactive visual interfaces.” [TC05]. The focus are interactive visualizations which allow users to steer the analytical reasoning process. This process is depicted in figure 2.1. Based on (transformed) data, models are generated automatically. These models are visualized. The user can interact with the visualization and her feedback is integrated back into the model.

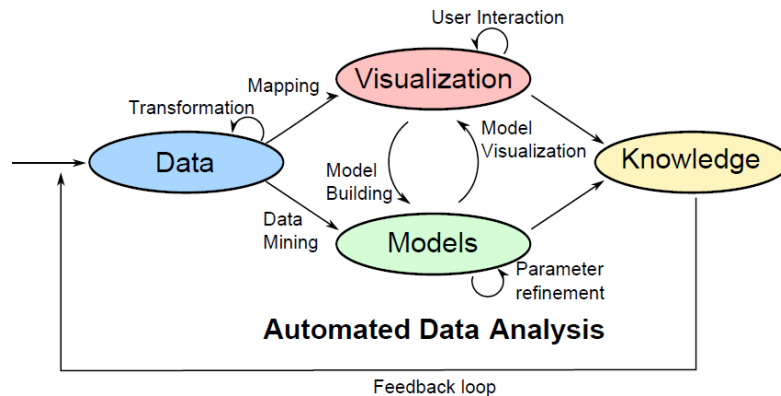


Figure 2.1: Overview of the Visual Analytics Process as defined in [KMOZ08]

The aim of Visual Analytics is to include human knowledge (with the help of interactive visualizations) into analytic models and thus be able to solve problems that would be unsolvable by either human or algorithms themselves. In other words, Visual Analytics

tries to combine the human knowledge and intuition with the power of computational models. Applications of Visual Analytics include advances in genomics (multi-scale information visualization) and analysis of massive unstructured text repositories [WT04].

Visual Data Mining [Ros06] has its origin in the the field of Data Mining and is sometimes interchangeably used with Visual Analytics, which has its origin in the field of Information Visualization. However, in Visual Data Mining, the algorithms used are mining algorithms, whereas Visual Analytics include all types of computational methods. Thus, Visual Analytics is the more general, more recent and more often used term.

The boundaries between Information Visualization and Visual Analytics are not clearly defined. Bob Spence expressed this fact in his keynote at the International Conference of Information Visualization 2011 about the relatively young discipline Visual Analytics as follows: “Visual Analytics is nothing new – I have been doing it for 40 years.” – meaning that, Visual Analytics is just another term for interactive visualizations with a special purpose.

In this thesis Visual Analytics is understood as an extension of Information Visualization focusing on interactive visualizations, whereas Information Visualization may also include non-interactive visualizations.

2.2 Supervised Learning – Classification

Classification in the scope of this thesis means the inductive supervised machine learning process. Inductive means, that the learner learns a general model from examples. Supervised learning means, that the learner gets the information about the target function along with the examples. Thus, in the case of classification the learner gets examples and associated class labels. Several types of classification problems can be distinguished, depending on the output and the input type, the task and so on. For more details see section 2.2.1.

A wide variety of classifiers exists, an example categorization can be found in [Seb02]. The author distinguishes the following categories: probabilistic classifiers, artificial neural networks, decision rule-based classifiers, instance-based learners and **Support Vector Machines (SVMs)**.

Probabilistic classifiers investigate the statistical distributions of attributes to predict the class label. A prominent example of probabilistic classifiers is the **Naive Bayes (NB)** classifier, see Section 2.2.4. **Artificial Neural Networks (ANNs)** try to model the functioning of the human brain. Decision-rule based classifiers include decision trees and try to learn simple rules from the data (in case of trees these rules are hierarchically related). Both, neural nets and decision trees are not commonly used in text classification, and therefore not further referenced in this thesis. Instance-based learners are represented by the **k-Nearest Neighbor (KNN)** algorithm described in section 2.2.3. **SVMs** are originally binary classifiers attempting to calculate a hyperplane in the high-dimensional

2 Foundations

space that best distinguishes positive and negative training samples. SVMs have been generalized to multi-class problems, a detailed description of the algorithm can be found in section 2.2.5.

Interestingly, three of the four classifiers used in this work (NB, SVM, and KNN) were identified amongst the 10 most influential algorithms in data mining by the IEEE International Conference in data Mining (ICDM) 2006 [WKRQ⁺08].

The choice of the classifier depends not only on the type of the classification problem (see sec 2.2.1). Restrictions imposed by the classification problems are for instance: multi-label algorithms can not be used for single-label or binary classification, but vice-versa is possible, since the multi-label case can be constructed from the single-label case, for details see the review in [dCF09]. Further the requirements of the application influence the choice of the classifier. Requirements of the application may be the runtime performance, storage restrictions, online-learning requirements, performance and the necessity to interpret the model. It has been shown empirically [Kot07] and theoretically [Wol96a] that no single classifier can outperform other algorithms over all data sets. The predictive performance of the classifier has to be estimated separately for each classification task (see section 2.4).

This section is structured as follows: First, a categorization for classification problems is described in section 2.2.1. Trivial classifiers are introduced as baseline for classifier comparison in section 2.2.2. Four different common classifiers are described in detail, KNN in section 2.2.3, SVM in section 2.2.5, NB in section 2.2.4, and Class-Feature-Centroid (CFC) classifier in section 2.2.6. These four classifiers are the main classifiers used in this thesis. Section 2.2.7 discusses the calculation of a-posteriori probabilities from general classifier outputs. The general framework for evaluating classifiers is presented in detail in section 2.4 for binary classification and multi-class classification problems. Multi-label and other classification problems are not covered here, because they are not in the focus of this thesis and not used in the experiments.

2.2.1 Types of Classification Problems

Classification problems can be categorized along various dimensions. Classification problems may be differentiated by the number of classes, the relation between classes, the number of labels assigned to one item, the type of the output, the type and the representation of the items' features, and the item type. An overview over these dimensions is given in table 2.1. For instance, classifying emails into the classes "spam" and "not spam" would be a binary, single-label text classification problem, the features can be numerical, and sparsely represented. Furthermore one can distinguish between offline and online learning. Offline learners are trained once with the complete training data set. Online learners are trained incrementally, and update their model when getting new training data. Online learning can be further differentiated into serial (one instance at a time) and batch learning (multiple instances at a time).

Table 2.1: Dimensions of classification problems

Dimension	Characteristics
number of classes	binary or multi-class
relation between classes	flat, hierarchical, arbitrary structure
number of labels	single-label or multi-label
input (feature type)	numeric, ordinal, ..
output	binary, ordinal or ranking; with or without confidence value; a-posteriori distribution
feature representation	sparse or dense
item type	e.g., images, texts, genes

The classification problems covered in this thesis are multi-class, single-label, flat with probabilistic output. Considered item types in the experiments are images and text, whereas for the image features a dense feature representation and for the text features a sparse representation is used.

2.2.2 Trivial Classifiers

The concept of trivial classifiers is used to better assess the performance of trained classifiers. Trivial classifiers do not have a model that was learned from the training data, but a fixed internal classification rule. For a so-called *trivial rejector* the internal rule would be to assign all items to the class "false" in the binary classification task. The *trivial acceptor* on the other hand, assigns all items to the class "true". Generalizations for multi-class problems are the *trivial majority classifier* and the *trivial minority classifier*. The former assigns all items to the most occurring class, the second to the least occurring class. Also of interest is the *random classifier* or *random guessing*, a classifier which assigns the items to the classes randomly, sampling either from a uniform distribution or based on the a-priori distribution of the class labels.

The actual value of the accuracy measure and therefore the quality of a classifier has no meaning until compared to the trivial cases. This comparison can either be done explicitly by denoting the performance measures for the trivial classifiers or implicitly by knowing the underlying data set and classification problem. The latter is mostly used in publications which use standard data sets.

I will give an example to illustrate the importance of the comparison to the trivial cases. A classifier that correctly classifies 82% of the data set seems to be a good classifier. 4 out of 5 items are classified correctly. However, in case of the Shuttle data set ² this is only marginally more than the trivial majority classifier would do, because 80% of

²[http://archive.ics.uci.edu/ml/datasets/Statlog+\(Shuttle\)](http://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle))

2 Foundations

the items belong to class 1. So actually, this classifier performs bad, a judgment that can only be made when either knowing the data set, knowing the performance measures for the trivial classifiers, or using alternative performance measures more suitable for skewed data.

2.2.3 K-Nearest Neighbor Classifier (KNN)

Nearest neighbor algorithms [CH67] are prominent examples for so called lazy learners. Lazy learners do not build an abstract model of the training data, but simply store it. The calculations are deferred to the classification time.

The KNN algorithm determines the k closest items in the training data and then decides the label based on the class labels of these neighbors. The decision for the label can simply be based on majority voting (see equation 2.1) taking the predominant label of the nearest neighbors as result.

$$y' = \arg \max_v \sum_{(x_i, y_i) \in N(x)} I(v = y_i) \quad (2.1)$$

$$I(a) = \begin{cases} 1 & \text{if } a \text{ is true} \\ 0 & \text{otherwise} \end{cases} \quad (2.2)$$

$N(x) \subseteq D_l$ is the set of identified nearest neighbors from the training set D_l for the test item x . y_i is the label of the training example x_i , v is the class label currently counted by the sum, and the function I indicates whether the current label y_i is the same as the class label under investigation v . The resulting decision is the label y' for the test item x .

An alternative decision for the label of an item x is based on distance weighting, which can improve the classifiers performance if the nearest neighbors vary widely in their distances. Distance weighting (see equation 2.3) multiplies the votes for each class label with a weight depending on the distance of the respective neighbor and thus makes the decision more insensitive to the choice of the parameter k . For the weighting function w_i usually the reciprocal of the squared distance is chosen.

$$y' = \arg \max_v \sum_{(x_i, y_i) \in N(x)} w_i \cdot I(v = y_i) \quad (2.3)$$

$$w_i = \frac{1}{d(x, x_i)^2} \quad (2.4)$$

An important choice to make is the distance function d . A prominent and widely applied distance measure is the Euclidean distance. However, when the number of attributes

increases, the Euclidean distance becomes less discriminating. Thus, for the task of text classification the cosine similarity is more appropriate. Another issue regarding distance functions are the difference in attribute ranges. Usually scaling the attribute ranges or weighting the attributes prevents them from unequally influencing the distance measure and consequently the classification result.

Another important choice to make is the choice of the basic parameter k . If k is chosen too small, the classifier becomes too sensitive to noise, if k is too large, too many irrelevant neighbors are found. Furthermore, for $k \geq |D_l|$, all training items become nearest neighbors and the **KNN** classifier degenerates to a simple trivial majority classifier when using majority voting (compare section 2.2.2).

The informative **KNN** [SHZ⁺07] is one extension of the basic algorithm aiming at finding the optimal k for a task at hand. Other extensions aim at reducing the number of stored training samples and thus the classification costs while retaining the predictive power of the algorithm, see for instance [Har68]. The class of instance-based learning algorithms extend the basic **KNN** focusing on reducing storage requirements [AKA91]. The advantages of the **KNN** algorithm are its fast training time, its easy understandability and its power to correctly classify items that are not linearly separable. **KNN** has successfully been applied to text categorization problems [Seb02].

2.2.4 Naive Bayes Classifier (NB)

The **NB** classifier is a probabilistic classifier based on the Bayes theorem. The **NB** classification scheme makes two assumptions about the data set: First, all attributes are independent of each other. Second, all attributes are equally important.

The **NB** classifier learns a model for the joint probability of the class label y and features f_i and makes its predictions by applying Bayes' rule to calculate the conditional probability for the class labels when given the features. In other words, the classifier learns a model for $p(y, f_1, \dots, f_n)$ and makes its predictions calculating the conditional probability $p(y|f_1, \dots, f_n)$.

$$p(y|f_1, \dots, f_n) = \frac{p(y) \cdot p(y|f_1, \dots, f_n)}{p(f_1, \dots, f_n)} \quad (2.5)$$

The **NB** classification model allows to calculate the probability of each class y given the values for the features f_i . To make the computation of the model feasible the conditional independence assumption $p(f_i|y, f_j) = p(f_i|y)$ is used. Applying the Bayes formula, the conditional independence assumption and the chain rule for conditional probabilities to

2 Foundations

equation 2.5 leads to the formula for the joint probability distribution in equation 2.6.

$$p(y, f_1, \dots, f_n) = p(y) \cdot p(f_1|y)p(f_2|y) \dots p(f_n|y) \quad (2.6)$$

$$= p(y) \prod_{i=1}^n p(f_i|y) \quad (2.7)$$

Thus, the class label can be estimated by the class with the highest conditional a-posteriori probability as given in equation 2.8. The calculation factors in the class prior $p(y)$ and the independent conditional probability distribution for each feature $p(f_i|y)$.

$$y' = \arg \max_y (p(y|f_1, \dots, f_n)) \quad (2.8)$$

As both assumptions, conditional independence and equal importance of attributes, are usually not met in real-life data sets, the **NB** classifier is often outperformed by other classification schemes. However, experiments have also shown, that the **NB** can outperform decision tree induction, instance-based learning and rule induction [DP97] on standard data sets. The advantages of the **NB** classifier are that it performs reasonable well even if little training data is given, it has a short training time and a straight-forward incremental version. Further, it is parameter-free and thus no model-selection step is needed. Important in the context of understanding is its easily interpretable classification scheme. **NB** classifiers are widely used in text classification and spam filtering.

2.2.5 Support Vector Machines (SVMs)

SVMs have been introduced by Vapnik [Vap95, Vap98]. A **SVM** aims at defining decision boundaries between classes in the high-dimensional space. In the linear, binary case, this decision boundary is a hyperplane given by equation 2.9 where $w \in \mathbb{R}^N$ and $b \in \mathbb{R}$. The decision function f in equation 2.10 then decides the class label depending on which side of the hyperplane the data point lies.

$$w \cdot x + b = 0 \quad (2.9)$$

$$f(x) = \text{sgn}(w \cdot x + b) \in \{-1, 1\} \quad (2.10)$$

For linearly separable data, so called hard-margin **SVMs** are applied. For a set of labeled data $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$ where $x_i \in \mathbb{R}^N$ and $y_i \in \{0, 1\}$, the optimal hyperplane is the hyperplane that separates positive and negative samples and maximizes the margin. This means the sum of the distance of the data points and the hyperplane is maximized while keeping negative and positive samples separated:

$$\max_{w,b} \left\{ \min_{x_i} \{ \|x - x_i\| : x_i \in \mathbb{R}^N, w \cdot x + b = 0 \} \right\} \quad (2.11)$$

Soft-margin classifiers are used when the data is not (linearly) separable. Soft-margin classifiers introduce a misclassification cost c for each misclassified training example. For details about how to solve the optimization problems refer to [DHS00].

The decision space of the SVM is characterized by the support vectors only, which makes SVM prone to errors. Another approach for nonlinear data is to apply a kernel function to map the input data in a higher-dimensional space where it will be linearly separable. However, this projection to a high-dimensional space increases not only the discriminating power of the SVM but also the training complexity. As the SVM is only directly applicable for binary problems, multi-class classification task must be split into multiple binary classification tasks (one-versus-all or one-versus-one).

For text classification linear SVMs have shown to achieve good performance and the use of kernels did not improve the performance [Joa98, DPHS98].

2.2.6 Class-Feature-Centroid Classifier (CFC)

The CFC classifier [GZG09] was especially designed for multi-class, single-label text classification problems. The centroid-based classifier applies a special centroid construction taking into account both, the inter-class term distribution and the inner-class term distribution. Both term distributions are then combined to generate the weight for the i -th term of centroid j as follows:

$$w_{ij} = b^{\frac{DF_{t_i}^j}{|c_j|}} \times \log\left(\frac{|c|}{CF_{t_i}}\right) \quad (2.12)$$

where $DF_{t_i}^j$ is term t_i 's document frequency in class c_j , $|c_j|$ is the number of documents in class c_j , $|c|$ is the total number of document classes (i.e. the total number of centroids). CF_{t_i} is the number of classes containing term t_i , and b is a constant, $b > 1$. This weighting scheme produces highly discriminant centroids, each of which represents a class.

A text document is then classified by labeling it with the class label (y') of the most similar class centroid. For computing the similarity of document vectors \vec{d}_i to class centroid vectors (\vec{c}_j) a denormalized cosine similarity (sim) is used. This similarity measure was chosen by the authors of the original paper, because it preserves the discriminant capabilities of the centroids. Thus, the final class label y' is computed from the distance of the document vector \vec{d}_i to all centroid vectors \vec{c}_j as follows:

$$y' = \arg \max_j (sim(\vec{d}_i, \vec{c}_j)) \quad (2.13)$$

$$sim(\vec{d}_i, \vec{c}_j) = \cos \vec{d}_i \cdot \|\vec{c}_j\|_2 \quad (2.14)$$

The CFC classifier is very fast in terms of training and classification, its training time complexity is linear and the classification complexity is constant in the number of classes.

Guan et al. [GZG09] report classification performance of the CFC algorithm comparable to SVM on standard text data sets.

2.2.7 A-Posteriori Probabilities

As described in section 2.2.1 the output of a classifier for a test item may either be a binary value, one or more class labels and confidence values for each label, or a-posteriori probability distribution over the class labels. In order to combine the outputs of different classifiers the outputs must be converted to comparable values, usually to a-posteriori probabilities. Also in the context of classifier visualization comparable outputs are desirable to design classifier-agnostic visualizations of classification results.

Usually SVMs output a binary decision: the item belongs to the class or it does not belong to the class. In the multi-class case the output is a class label. Additionally to the class decision SVMs output the distance of the data item to the decision boundary. This distance is an uncalibrated value that is not a probability. An intuitive means for assigning probabilities to data points classified by a SVM was introduced by Platt [Pla99]. All examples are projected onto an axis perpendicular to the hyperplane and then logistic regression is performed to extract class probabilities.

Duin and Tax [DT98] summarize and compare different methods to calculate posterior probabilities depending on the type of the classifier. Bayes estimation is used for density-based classifiers, e.g KNNs and Decision Trees. For classifiers outputting a distance to the boundary as SVM and Linear Discriminant Analysis (LDA) the authors propose fitting of a sigmoidal function as described above. Sigmoidal functions for estimating a-posteriori probabilities can have also been applied to hierarchical classification [Gra03] (on page 43).

2.2.8 Summary

The choice of classifier depends on the type of classification problem, see section 2.2.1 and on the task at hand. Kotsiantis et al. [Kot07] compared various classifiers by training time, incremental, missing values, noise tolerance, number of parameters, and interpretability. Four classifiers were described in this chapter, namely KNN (section 2.2.3), SVM (section 2.2.5), NB (section 2.2.4) and CFC (section 2.2.6).

The no-free-lunch theorems [WM97] state that no single algorithm outperforms all others on all data set. Sometimes random guessing might work best on the test data set. Thus, performance estimation is used to find the most appropriate classifier. However, one needs to carefully interpret the various available performance measures. Performance evaluation for classifiers is described in section 2.4.

2.3 Text Classification

This section briefly describes the general approach and the specific feature used for text classification.

A text classifier takes as input a document d and outputs a category from a predefined set. The learning algorithm for a text classifier takes as input a set of labeled documents D_l and outputs a trained classifier. The general types of classification, e.g., single- vs. multi-label as defined in section 2.2.1 apply to text classification. The specialty of text classification is the representation of the documents. In the experiments of this thesis the vector-space representation of documents is applied.

The feature generation for text classification consists of two steps: (i) text preprocessing, and (ii) vectorization. For the preprocessing step we give the full possible pipeline here, however, depending on the problem single steps, e.g. [Part-of-speech \(POS\)](#)-tagging, may be unnecessary and can be omitted. Figure 2.2 summarizes the text classification pipeline.

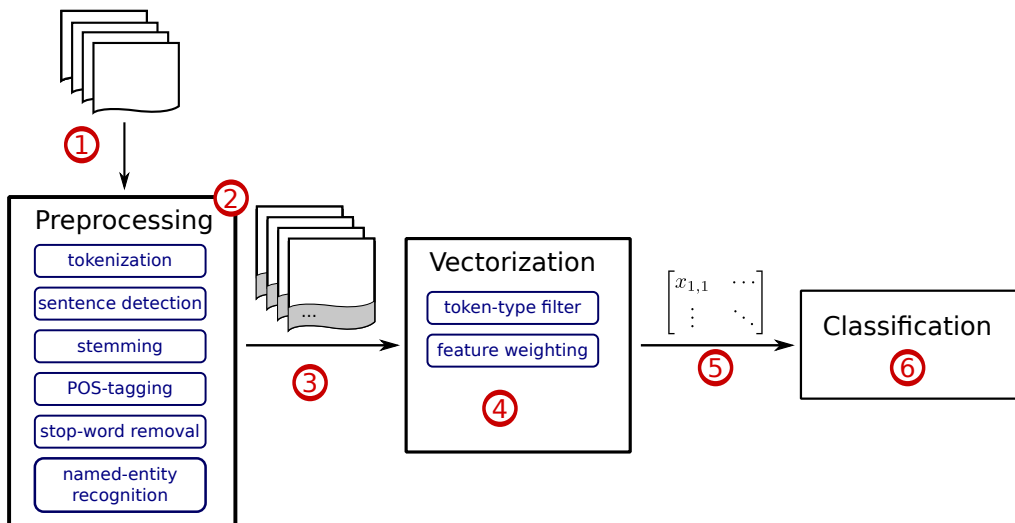


Figure 2.2: Overview of the text classification pipeline. ①The input is a set of documents. ②On the documents various preprocessing steps are applied. ③The output of preprocessing step are annotated/enriched documents. ④The vectorization step converts the annotated documents to a feature matrix, ⑤which is the input for the classification ⑥.

Preprocessing: First, the document is tokenized, i.e. separations between words and punctuations are marked. Second, start and endpoints of sentences are detected. These two steps are mandatory for text classification. Third, stop-words, e.g., “and”, “or”, are marked or removed. The rationale for stop-word removal is that they do not contribute

2 Foundations

to the information in the document, but rather improve fluency and readability of the text. Fourth, words are stemmed, i.e. replaced by their word-stem, i.e. “originally” and “originates” are stemmed to “origin”. Fifth, the grammatical function of the tokens is detected and annotated with so-called **POS**-tags. Now we know for each token whether it is a noun, an adjective, a punctuation and so on. The so enriched document serves as input for the vectorization step. Optionally, named entities, such as names of persons, or organizations, can be recognized and marked in the document.

Vectorization: In the vectorization step, a vector-space is generated from the enriched documents. In the simplest case, each unique token (term) constructs one dimension of the vector-space. The mapping between the tokens and the dimension in the vector-space is called the dictionary. Each document is then represented as a vector in the vector space as follows: For each document the occurrences of each token are counted and this count, the so-called term frequency, is assigned to the dimension of the vector that corresponds to the token. Optionally, one might choose to only use nouns or punctuation to generate the vector-space and apply different feature weighting schemes. In this thesis two different weighting functions are used, namely **Term Frequency - Inverse Document Frequency (TF-IDF)** [MRS08] weighting and **Okapi BM25 ranking (BM25)** [JWR00].

The **TF-IDF** weight of a term t for a document d is defined as the product of the term’s frequency ($\text{tf}_{t,d}$) in this document and the term’s inverse document frequency (idf_t) as:

$$\begin{aligned}\text{tf-idf}_{t,d} &= \text{tf}_{t,d} \cdot \text{idf}_t \\ &= \text{tf}_{t,d} \cdot \log\left(\frac{N + 1}{\text{df}_t + 1}\right)\end{aligned}$$

where N is the total number of documents in the corpus and df_t the number of documents containing term t . In short, this means a term gets a large weight for a document d if it is contained in only few documents and occurs often in d . On the contrary, a term is weighted low for a document, if it is contained in many documents and has very few occurrences in d .

BM25 does refer to a whole family of functions, the most commonly **BM25** weighting function is defined as follows for term t in document d :

$$\text{bm25}_{t,d} = \frac{(k_1 + 1) \cdot \text{tf}_{t,d}}{k_1 \left((1 - b) + b \cdot \left(\frac{L_d}{L_{avg}} \right) \right) + \text{tf}_{t,d} \cdot \text{idf}_t}$$

where L_d is the length of document d and L_{avg} is the average document length in the corpus. k_1 and b are tuning parameters, scaling the influence of document length normalization and the document term frequency respectively. idf_t is the inverse document frequency of term t as above and $\text{df}_{t,d}$ is the frequency of term t in document d .

To give an example for the terminology, the expression “noun-vector space with **TF-IDF** weighting” refers to a vector-space created from all words **POS**-tagged with “noun” using the **TF-IDF** weighting function.

Because some of the experiments in this thesis apply classification for a cross-domain task, this term will be explained here. A *cross-domain classification* task is a task where the training set to build the classification model has other characteristics (another underlying data distribution) than the documents the trained classifier will be applied to. One reason for training a classifier on another domain is for instance the availability of training data. For example, in the experiments of this thesis various classifiers are trained on news articles and evaluated on blog posts.

For more details about text classification refer to [Seb02, MRS08].

2.4 Performance Evaluation for Classifiers

In this section an overview of commonly applied performance measures for evaluating single-label classifiers are presented. The contingency table, which is the basis of the definition of the performance measure is introduced in section 2.4.2. The derived measures are then summarized in section 2.4.3 for binary classification problems and in section 2.4.4 for multi-class problems. Finally, a summary is given in section 2.4.5.

In general, as stated by the no-free-lunch theorems [Wol96a, DHS00] there is no single classifier that is known a-priori to perform best for a given task. In fact, there is no single classifier that is guaranteed to outperform random guessing on an arbitrary classification task. That means, the performance of a classifier must be evaluated for each learning problem (data set, prior knowledge) at hand. To judge the performance of a classifier the so called off-training error is used, i.e., the error (or another performance measure) for items that are not part of the training set. The main reason for using the off-training error is that a sufficiently powerful classification algorithm can learn the training set perfectly. A perfectly learned training means not necessarily a perfect behavior on off-training items, on the contrary, it could indicate overfitting. A classifier is said to overfit the training data if it performs well on the training data but poorly on the off-training data. The classifier is then said to have a poor generalization performance. Methods to estimate classifier performance including cross-validation are described in section 2.4.1.

The following sections assume items from an off-training set, either a separate evaluation set or a split of the training set that was not used to train the algorithm.

2.4.1 Evaluation Methods

To assess the performance of a classifier the evaluation must be performed on data that has not been used for training. Otherwise the classifier might seem to perform very well (high accuracy on the training set), but actually has not learned a model from the data. It only memorized the data and thus generalizes badly to new data.

There are three main methods to estimate the performance of a classifier. First, by providing a separate, labeled data set (evaluation data set) and calculate the measure

2 Foundations

on this set. Second, by splitting the training set to create the evaluation set. These methods are valid if it can be guaranteed that the evaluation data set is a representative sample of the whole data set, which is usually not the case. The third method is called cross-validation. Cross-validation is used if the training set is relatively small, and labeling new items is costly. The training set is split into n subsets ($n \in \{5, \dots, 10\}$ in practice). The classifier is trained and evaluated n -times (runs) whereas the i -th split is used as evaluation split and all other splits are used for training. The performance is then averaged over all runs. Cross-validation is the state-of-the-art method to evaluate classifiers. But, even with cross-validation one might not be able to correctly assess and compare classifier performance. Raeder et al. [RHC10] have shown, that classifier performance varies depending on data set, cross-validation method (number folds) and the random seed for the specific fold. The authors propose to perform multiple cross-validation runs to tackle this problem.

2.4.2 The Contingency Table

The contingency table, also called confusion matrix is a tabular representation of different types of errors produced by a classifier. In this table the desired output is compared to the prediction made by the classifier. The desired output is the correct label of the item, the so called actual class. The information of all actual classes is called ground truth. The predicted class is the class label the classifier assigned to a specific item. The predicted class can and mostly does vary for different classifiers. The actual class is invariant to specific classifiers because it represents the real-world. Table 2.2 shows a prototypical contingency table for binary classification problems. For binary classification problems four different cases can be distinguished: (i) Both, the actual and the predicted class is “true”. Such items are counted as **True Positives (TP)**. (ii) Both, the actual and the predicted class is “false”. Such items are counted as **True Negatives (TN)**. (iii) The actual class is “true”, but the prediction is “false”. Such items are counted as **False Negatives (FN)**. It means the item is classified as “false” (negative), but this prediction is incorrect. (iv) The actual class is “false”, but the prediction is “true”. Such items are counted as **False Positives (FP)**, meaning the item is classified as “true” (positive) but this prediction is incorrect. In some application domains the prediction “true” is also called an accept, whereas the prediction “false” is called a reject. Consequently, a **TP** is a correctly accepted item, and **FP** an incorrectly accepted one.

Table 2.2: Contingency table for binary classification problems.

		actual Class	
		true (positive)	false (negative)
predicted class	true (positive)	True Positives (TP)	False Positives (FP)
	false (negative)	False Negatives (FN)	True Negatives (TN)

Based on the contingency table the following performance measures can be defined (amongst others).

- accuracy a :** Rate of correctly classified examples ($TP+TN$). The number of correct classifications divided by the total number of samples. $a \in [0, 1]$.
- error e :** Rate of incorrectly classified examples. The number of incorrect classifications divided by the total number of samples. Also called the Zero-One-Loss $e \in [0, 1]$.
- Precision π :** Ratio of correct positive predictions (TP) to the total number of positive predictions. $\pi \in [0, 1]$.
- Recall ρ :** Ratio of correct positive predictions (TP) to the total number of positive samples. Also called sensitivity. $\rho \in [0, 1]$.
- Specificity s :** Rate of correct rejections (TN) to the total number of negative samples. $s \in [0, 1]$
- F-measure F_β :** Averages precision and recall, prefers precision for $\beta > 1$. Usually the F_1 -measure is used, which is the geometric mean of precision and recall.

The calculation of these measures is straight-forward for binary classification problems, but more complex for multi-class problems. Both cases are described in detail in the following sections.

2.4.3 Binary Classification

The performance measures introduced in section 2.4.2 can be directly derived from the contingency table shown in table 2.2 as follows:

$$a = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.15)$$

$$e = 1 - a \quad (2.16)$$

$$\pi = \frac{TP}{TP + FP} \quad (2.17)$$

$$\rho = \frac{TP}{TP + FN} \quad (2.18)$$

$$s = \frac{TN}{FP + TN} \quad (2.19)$$

$$F_\beta = \frac{(\beta^2 + 1) \cdot \pi \cdot \rho}{\beta^2 \cdot \pi + \rho} \quad (2.20)$$

Less commonly applied performance measures, e.g. precision-recall-break-even and coverage are listed in [Seb02, KP98]. Further, Sokolova et al. suggested to use measures

commonly applied in medical diagnosis, e.g., Youden’s index and discriminative power for classifier evaluation [SJS06]. Recently also aggregated measures have been proposed, for instance the relative performance metric (RPM) [SKVH09]. However, most commonly applied measures in the scientific community are accuracy, precision and recall.

2.4.4 Generalization to Multi-Class Problems

There are two ways of generalizing the performance measures from binary classification problems to multi-class problems, namely based on (i) contingency-tables or (ii) class confusion matrices. In the first method, the contingency tables for all classes are calculated, one for each class. Each classification of a sample is then counted c -times (c - number of classes). For instance, an item with actual class C_1 and predicted class C_2 is counted as **FN** for class C_1 , as **FP** for class C_2 and as **TN** for all other classes $C_i, i \neq 1, 2$. The final performance measures can be either calculated for each class and then averaged or can be calculated over the sum of all decisions (see for instance [Seb02]). The former is called micro-averaging and the latter is called macro-averaging. Macro-averaging uses a so-called global contingency table derived as the sum of the contingency tables for all classes. The difference between micro- and macro-averaging is, that micro-averaging equally weights each sample, but implicitly gives higher weights to classes with many samples (item-pivoted measure), whereas macro-averaging equally weights each class (class-pivoted measure). The following formulas show the calculation of the micro- and macro-averaged performance measures, wheres micro-averaging is indicated by a superscripted μ and macro-averaging is indicated by a superscripted M .

$$a^\mu = \frac{\sum_i TP_i + \sum_i TN_i}{\sum_i TP_i + \sum_i TN_i + \sum_i FP_i + \sum_i FN_i} \quad (2.21)$$

$$\pi^\mu = \frac{\sum_i TP_i}{\sum_i TP_i + \sum_i FP_i} \quad (2.22)$$

$$\rho^\mu = \frac{\sum_i TP_i}{\sum_i TP_i + \sum_i FN_i} \quad (2.23)$$

$$a^M = \frac{1}{c} \cdot \sum_i a_i \quad (2.24)$$

$$\pi^M = \frac{1}{c} \cdot \sum_i \pi_i \quad (2.25)$$

$$\rho^M = \frac{1}{c} \cdot \sum_i \rho_i \quad (2.26)$$

The above formulas can be simplified for the classification problems discussed in this thesis, namely single-label multi-class problems. For these problems $\sum_i FP_i = \sum_i FN_i$. This is because for each prediction which does not equal the actual class, the sample is counted once as **FN** for the predicted class and once as **FP** for the actual class. From this follows that $\pi^\mu = \rho^\mu = F_1^\mu$ for the single-label case. The next subsection shows the results of micro- and macro-averaging for a small example data set.

The second method of generalizing performance measures is only applicable to accuracy, error and micro-averaging, but commonly used in evaluation of classifiers [WF05]. The method uses the class confusion matrix. The class confusion matrix is a generalization of the contingency table for multiple classes. The rows of the matrix represent the prediction, the columns represent the actual classes. Each cell then contains the number of items actually belonging to the class of the row for which the prediction of the column was made. A general class confusion matrix is shown in table 2.3. The method based on

Table 2.3: General class confusion matrix for three classes A, B and C and all possible predictions A', B', C'. A'|B means that B is the actual class and A' was predicted by the classifier.

	actual A	actual B	actual C
predicted A'	A' A	A' B	A' C
predicted B'	B' A	B' B	B' C
predicted C'	C' A	C' B	C' C

the class confusion matrix only differentiates correct and incorrect classifications. This viewpoint reduces the problem to a binary decision (correct/incorrect). The classification error is then calculated as the total number of misclassifications divided by the total number of classifications. Similarly, the accuracy is calculated as the total number of correct classifications divided by the total number of classifications. In terms of the class confusion matrix, accuracy a is the ratio between the sum of all diagonal elements and the total sum of elements, whereas the classification error e is the the sum of all off-diagonal elements divided by the total sum of elements. Micro-precision π^μ and micro-recall ρ^μ can be calculated from the class confusion matrix with $\sum_i TP_i$ being the sum of all diagonal elements in the confusion matrix, and $\sum_i FN_i = \sum_i FP_i$ being the sum of all off-diagonal elements. In contrast to the first method, where each sample is counted once for each class, here each sample is counted only once (either as correct decision or as incorrect decision).

Example

This section discusses the calculation of the above mentioned performance measures for a simple classification problem. One of the three classes A, B and C should be assigned to each of the data items. The classifier to be evaluated has already been trained. The output of the classifier and the ground truth are summarized in table 2.4. Tables 2.5 show the contingency tables for each of the classes and the micro-averaged contingency table and table 2.6 shows the confusion matrix. Using the formulas for the multi-class problems, the performance measures are calculated as in equations 2.27 to 2.32. Note: error and accuracy without a superscript notation are calculated from the class confusion matrix.

2 Foundations

Table 2.4: Ground truth and predictions of a hypothetical classifier for a classification problem with three classes

	label (predicted)	target (actual class)
item 1	A	A
item 2	B	B
item 3	C	C
item 4	A	B
item 5	A	C

Table 2.5: Class contingency tables for classes A, B and C and the aggregated (micro-averaged) contingency table for example in table 2.4

class A			class B			class C			micro-averaged		
	true	false		true	false		true	false		true	false
true	1	2	true	1	0	true	1	0	true	3	2
false	0	2	false	1	3	false	1	3	false	2	8

Table 2.6: Class confusion matrix for example in table 2.4

class confusion matrix			
	actual A	actual B	actual C
predicted A	1	0	0
predicted B	1	1	0
predicted C	1	0	1

$$e = \frac{2}{5} = 0.4, \quad a = \frac{3}{5} = 0.6 \quad (2.27)$$

$$a^\mu = \frac{11}{15} \approx 0.733, \quad a^M = \frac{1}{3} \cdot \left(\frac{3}{5} + \frac{4}{5} + \frac{4}{5} \right) \approx 0.733 \quad (2.28)$$

$$\pi^\mu = \rho^\mu = F_1^\mu = \frac{3}{5} = 0.6 \quad (2.29)$$

$$\pi^M = \frac{1}{3} \cdot \left(\frac{1}{3} + 1 + 1 \right) \approx 0.778 \quad (2.30)$$

$$\rho^M = \frac{1}{3} \cdot \left(1 + \frac{1}{2} + \frac{1}{2} \right) \approx 0.667 \quad (2.31)$$

$$F_1^M = \frac{1}{3} \cdot \left(\frac{1}{2} + \frac{2}{3} + \frac{2}{3} \right) \approx 0.661 \quad (2.32)$$

2.4.5 Summary

As outlined in this chapter there is a wide variety of performance measures for evaluating classifiers. Only the most commonly used were mentioned in the above sections. Additionally, there are at least three ways of generalizing from measures defined in the binary case to multi-class problems. This means, in order to interpret and compare classifiers it is important to know how the performance values were calculated (is the accuracy macro-averaged, micro-averaged or calculated from the class confusion matrix). Furthermore, because there is no single standard value with straight-forward interpretation it is not easy for non-expert users to assign the performance of classifiers. It is also important to consider the data set on which this measures were calculated, whether it was a separate evaluation split or cross-validation was performed. But even with the state-of-the-art approach cross-validation results might be misleading (see section 2.4.1). Also, as outlined in section 2.2.2 it is necessary to consider the trivial classifiers in order to assess the performance of classifiers.

It can be concluded, that assessment of classifier performance based on evaluation measures is difficult, even for experts.

2.5 Active Learning

Active Learning is a method to tackle the problem of too little training data. In an active learning scenario, a learner gets presented new training data which were selected by a sampling algorithm and labeled by a so-called oracle. The aim of active learning is to reduce the number of training samples while retaining the same performance of the learning algorithm. This is done by intelligently selecting the next training sample such that they provide as much as possible new information to the learning algorithm. The general active learning process is depicted in figure 2.3.

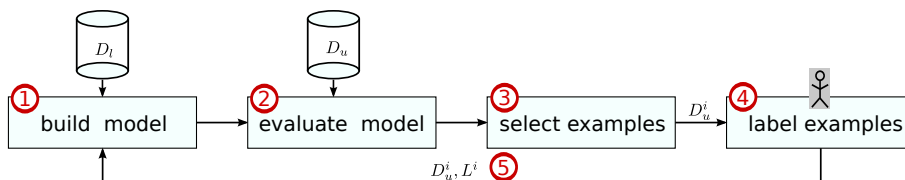


Figure 2.3: The active learning scenario. ① A classification model is built on labeled data. ② The classifier is evaluated on the unlabeled data. ③ An oracle selects the next items to label. ④ A human labeler assigns the labels. ⑤ The newly labeled instances serve as training data for the new classifier. The process continues in a loop.

Active learning approaches mainly differ in their selection strategy – the way the oracle selects unlabeled samples to be labeled. A straight-forward approach that is independent of the classifier is so called uncertainty sampling. The idea of uncertainty sampling is to

2 Foundations

chose the example of which the classifier is most uncertain about, expecting that by labeling it the most possible information is added to the classification model. The uncertainty can be measured in different ways. In this thesis two uncertainty sampling methods will be applied, maximum-entropy sampling and minimum-maximum confidence selection.

Maximum-Entropy Sampling From the set of unlabeled samples D_u the sample x_c with the minimal entropy in the a-posteriori distribution is selected:

$$x_c = \arg \max_x \left(- \sum_i (P(y_i|x) \cdot \log(y_i|x)) \right), \quad x, x_c \in D_u \quad (2.33)$$

Minimum-Maximum Confidence Sampling From the set of unlabeled samples D_u the sample i which has the least maximum value in the a-posteriori distribution is selected:

$$x_c = \arg \min_x \left(\arg \max_y P(y|x) \right), \quad x, x_c \in D_u \quad (2.34)$$

Further, *batch* and *serial* active learning are distinguished. In batch active learning multiple samples are selected at once in each selection step and presented to the classifier to update its model. In serial active learning one item at a time is selected and the classification model is updated instantly. In the experiments we use serial active learning, since batch active learning combined with uncertainty sampling has been shown to perform badly [Set10].

Settles [Set10] presents a survey also covering classifier-dependent active learning methods and empirical results.

Note, that the training set generated with a specific active learning algorithm (classifier and selection strategy) is inherently coupled to the active learner. This is because, the training set is built to optimize the current classifier. However, if classifiers are switched (to newer state-of-art classifiers, to classifiers more understandable to users), the training set may not be as useful anymore.

2.6 Voronoi Diagrams

This section briefly introduces the foundations of Voronoi diagrams as needed in this thesis.

A Voronoi diagram is a separation of a two dimensional space into several regions. Each region is defined by a single point, the so-called site or generator point. A region of a generator point p_i consists of all points in the space that are nearer to p_i according to a given distance measure than to any other point p_j , $i \neq j$. The Voronoi diagram is fully defined given a set of points $P = \{p_1, \dots, p_n\}$ and the description of the space

(including the distance metrics). Figure 2.4 shows an example of a Voronoi diagram with 11 generator points in the two-dimensional Euclidean space. For an extensive discussion on Voronoi diagrams see [OBSC00, AK00].

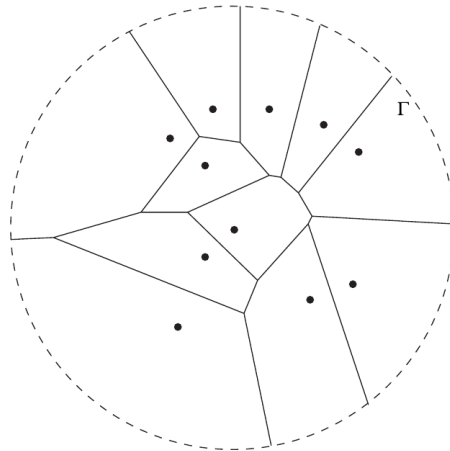


Figure 2.4: Voronoi Diagram in the two-dimensional Euclidean space with 11 generator points. Figure taken from [AK00].

2.7 Summary

This chapter introduced the foundations necessary to understand this thesis. First, the terms Information Visualization, Visual Analytics and Visual Data Mining were defined. Second, an overview of classification, evaluation measures and evaluation methodology is given. Further, the classification algorithms used in this thesis were presented in detail. Moreover, the general text classification pipeline was outlined. The concept of active learning described will later be extended using a visual approach. Finally, a brief introduction into Voronoi diagrams was given.

He who loves practice without theory is like the sailor who boards ship without a rudder and compass and never knows where he may cast.

(Leonardo da Vinci)

3 Theory

In this chapter the theoretical foundations of this thesis are developed, concrete implementations are described in the next chapter.

Remember, the research question of the thesis

RQ “Can interactive visualization improve construction, understanding, assessment, and adaptation of supervised machine learning algorithms?”

We identified five different levels of understanding of and user feedback to classifiers, namely the data set level, the class level, the item level, the feature level and the classifier level. These levels are summarized in table 3.1. More specifically, aspects that can in principle be **understood** about a classifier include (i) how well the classifier performs on the given data, which decisions it makes, what the basis for its decisions are, (ii) the distribution of the data items (linearly separable, topic overlap, two distinct subtopics integrated in one), (iii) dependencies of attributes, (iv) redundancy of attributes, (v) attribute relevance in the model. Some aspects may be communicated to machine learning experts by specifying the classification algorithm, the training parameters, the statistics of the training data and tables of evaluation measures. However, it is not feasible to communicate with lay persons in the same way and also for experts the visualizations can be a supportive tool. Aspects that can in principle be **feed back** to the classifier include (i) correct or approve label, (ii) identify and remove outliers, (iii) modify the relevance of attributes, (iv) remove attributes, (v) define interesting classes.

Assessment and Understanding: In general, visualizations for classifiers can be either classifier-agnostic or classifier-dependent. Classifier-agnostic visualizations are visualizations that can be applied to more than one or all classification algorithms. On the contrary, classifier-dependent visualizations are designed for one specific classification algorithm. In general, a classifier-dependent visualization can be tightly coupled to the model and therefore reveal more details about the classifier. On the other hand, a classifier-agnostic visualization would need to be designed only once, and would allow for comparison amongst different classifiers. Further, users of the visualization would only need to learn to interpret one visualization. Because of the latter reasons in this thesis the focus lies on classifier-agnostic visualizations.

Table 3.1: Aspects of classifiers, classification models and results communicated from classification models (understanding) and to classification models (feedback) on the level of items, the classes, the data set, features and the classifier.

Level	Aspects communicated	
	From the Model (Understanding)	To the Model (Feedback)
Data Set	number of samples class distributions performance measures for classifier performance compared to trivial cases	add, remove, split, merge class(es) add and remove items
Class	performance for each class conflicts between classes	
Item	assigned label confidence of the decision items hard to classify similar items	correct/approve label remove outliers
Feature	relevance of features for decision ambiguous features dependency of features	select key features for classes remove irrelevant/ambiguous features indicate dependency of features weight features
Classifier (specific)	decision boundaries boundary items ..	

As detailed in table 3.1 there are five levels of understanding a classifier:

1. data set level
2. item level
3. class level
4. feature level
5. classifier level

In the following it is argued (in section 3.1.1) that only the first three levels (data set, class and item) can be visualized in a classifier-independent manner. The properties of such a visualization are derived in section 3.1.2. Reviewing the state-of-the art in section 3.1.3 points toward two different types of visualizations covering the item level and the class level. An explicit discussion of the data set level will be omitted, because all of its aspects (performance measures, number of items, class distribution) are already commonly used in classification processes by either printing the specific values or using

well-known visualizations, such as histograms. Although the data set level is not discussed in detail, all of its aspects are displayed in the prototypical application without further being mentioned.

Visualizing the classifier level (decision boundaries, boundary items) is already well covered in the literature for almost all classifiers and is thus not the focus of this thesis. An overview of existing literature is given in table 3.2 in section 3.1.3.

The feature level can only be visualized in a classifier-dependent manner. Therefore, we chose a specific classification problem, namely text classification, and one specific classifier to show what can be done regarding assessment and understanding on the feature level. The theory towards this visualization is discussed in section 3.3.

Thus, at this point in discussion, we have the following state for the assessment and understanding part of the research question:

1. data set level, classifier-agnostic → existing visualizations, implemented, but not mentioned further
2. item level, classifier-agnostic → visualization will be designed (Class Radial Vis)
3. class level, classifier-agnostic → visualizations will be designed (Class Confusion Maps), some aspects also conveyed by Class Radial Visualization
4. feature level, classifier dependent → visualization will be designed for text classification problem (Voronoi Word Cloud).
5. classifier level → covered in literature, not focus of the thesis

Until now only the assessment and understanding parts of research questions were discussed. Naturally, assessment and a basic understanding is necessary for adapting a classifier.

Adaptation: For the same reasons as discussed in the assessment and understanding part we focus on classifier-agnostic adaptation of classifiers. This limits the possible influence on classifiers to correcting decisions that are obviously wrong. Using the levels described above, this kind of adaptation belongs to the item level. This seems a quite simple concept, but there is more to it. From the point of classifier performance it is crucial which decisions are corrected – some may not influence the classifier at all, some may have a greater influence on the model and its performance. Correcting decisions can be viewed as the generation of new training data. A classical approach to efficiently achieve this is Active Learning. We will develop a corresponding visual concept, called Visual Active Learning, which grants more influence to the user on the training data generation process. The concept of Visual Active Learning is outlined in section 3.4 by reviewing classical active learning, arguing differences and identifying properties of appropriate visualizations.

Construction: Regarding the construction part of the research question, we focus on a bottleneck in constructing classifiers that has been identified during the experiments performed for this thesis. Namely, for text classification tasks, the time for generating training data depends to a large degree on the time the human labelers need for compre-

3 Theory

hending the texts. Thus, we investigated whether an alternative representation of texts would speed-up the labeling process. These alternative text-presentations are developed in section 3.2.

Summing up the necessary theoretical ingredients for this thesis, the outline of this chapter is as follows: The theoretical foundations for classifier-agnostic visualization covering class and item level are described in section 3.1. The alternative text-presentations for faster text-comprehension in the training data generation step are developed in section 3.2. The theoretical foundations for classifier-dependent visualization for text classification covering the feature level are identified in section 3.3. The theoretical concept of Visual Active Learning is developed in section 3.4.

3.1 Towards Visualizations for Arbitrary Classifiers

In order to derive a visualization for arbitrary classifiers, it will be investigated in section 3.1.1, what classifiers do have in common, because only these common parts can be represented in a classifier-agnostic way. Second, from the identified common properties of classifiers, the desired properties of a classifier-agnostic visualization are discussed in section 3.1.2. Third, in section 3.1.3 the state-of-the-art literature is reviewed focusing on the suitability of available visualizations for the topic of this thesis. Finally, knowing the state-of-the-art and the desired properties, section 3.1.4 summarizes the gap in state-of-the-art which lead to the design of the visualizations described in the next chapter.

3.1.1 Common Properties of Classifiers

Metaphorically speaking, a classifier is just a box as in figure 3.1a where one puts something in, which is somehow transformed into something different. The transformation rule is either given or has to be learned beforehand. However, this is only the application view – with the input of the classifier being some data and the output being any or more categories. Without loss of generality, this discussion focuses on single-label classification (see section 2.2.1), so we assume to have only one category as output.

Properties of Classifiers: Going one step further, a classifier consists of a training algorithm and a classification algorithm as depicted in figure 3.1b. The training algorithm gets some specific kind of data – the training data – as input and generated a hypotheses from the data. The classification algorithm then uses this hypothesis to make predictions on new, previously unseen data – the test data. Learning the hypothesis is also called the training phase and doing predictions on new data is called the classification phase. Usually, the training algorithm and the classification algorithm are tightly coupled. This means, a [Support Vector Machine \(SVM\)](#) classifier can only apply hypothesis learned by a [SVM](#) training algorithm. Due to this tight coupling, sometimes, this division into training algorithm and classification algorithm is ignored and the unity of both is called the classifier or classification algorithm while knowing that it has to be trained first.

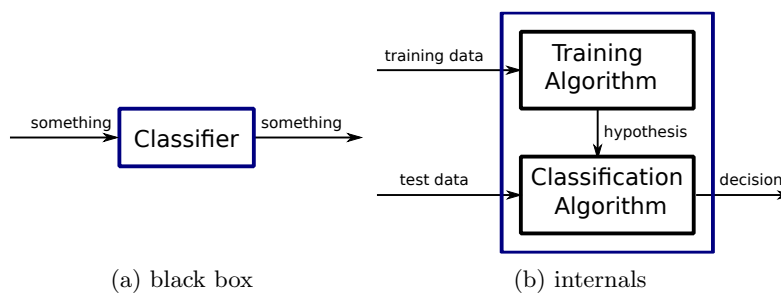


Figure 3.1: Views of a classifier. Black box (application) and internal (algorithmic) view.

3 Theory

Depending on the classifier, the learning algorithm generates different hypotheses classes. An examples is shown in figure 3.2. In this example, the data items have only two features, making the data visualization in the two-dimensional space feasible. Even a single classifier may generate different hypotheses classes, see for instance the SVM classifier in figure 3.2c (linear) and in figure 3.2d (Gaussian kernel). The different hypotheses classes for single classifiers are another argument for classifier-agnostic visualizations, aside from the already discussed arguments (only one visualization to design for experts and learn for users, one visualization allows for comparison).

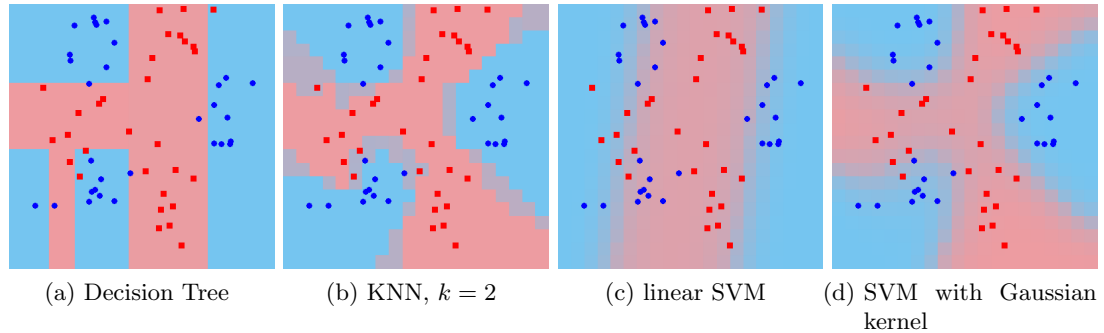


Figure 3.2: Example classifier hypothesis for 30 points of class 1 (blue circles) and 30 points of class 2 (red squares) in the 2-dimensional space. Pictures generated using the LocBoost applet¹.

Properties of the Data: In principle, a classifier can classify any general object or entity if the object is represented in a suitable form. Usually, the data is represented in form of a table (or matrix) where each row corresponds to a data item and each column corresponds to a feature. The process from a general object to the feature representation is called feature generation or feature engineering. Feature generation is not the focus of this thesis, the important point here is, that any data can be represented by feature vectors which serve as input for classifiers. Of course, depending on the input data, the feature vectors may have different dimensions and have different values (e.g. binary features, real-valued features) – but all of them can be represented as a number $\in \mathbb{R}$. Data items that are explicitly given a relation to other items are used in statistical relational learning, which is beyond the scope of this thesis.

The difference between the training data and the test data is: the training data comes with a label for each data item. This label is also called the ground-truth for an item.² Thus, in the training phase the learning algorithm gets presented a set of data item – label pairs. In the classification phase the input data has no label assigned, but the classifier should decide the label. One might know the ground-truth label for test items and use them to compare the output of the classification with the true label. By judging

¹<http://www.cs.technion.ac.il/~rani/LocBoost/>

²There are classifiers that can deal with weighted input data, but thats not the focus of this thesis.

the difference in the true label and the classifier's prediction one can get an estimate of the classifier's performance for unseen data.

Many classifiers not only output the predicted label but also a confidence value with the label. The confidence value is an estimate of the classification algorithm itself on how good the prediction is. More generally, the classifiers not only output one label and its confidence, but all possible class labels and for each of it a confidence value. Subsequent post-processing steps are applied to select the most probable label [KHDM98]. For classifiers, such as the [Naive Bayes \(NB\)](#) classifier, the labels and confidences are already an a-posteriori distribution over the classes, for other classifier the outputs can be mapped to a-posteriori distributions as described in section 2.2.7.

3.1.2 Properties of a Classifier-Agnostic Visualization

In the previous section the need for classifier-agnostic visualizations and following this way, common properties of classifiers were discussed. The following list summarizes, what a classifier-agnostic visualization must provide in order to aid in construction, understanding, assessment, and adaptation of classifiers.

1. To be applicable to a wider variety of classification problem the visualization should support multiple classes (not only binary, but **multi-class** classification).
2. Since **a-posteriori probabilities** are available or can be constructed from almost all classifiers, the visualization should use this information.
3. The classifier's **final decisions** and **confidences** should be shown. The confidences give the user an idea on how sure the classifier is about its decision and is necessary to thoroughly judge its behavior.
4. All **classes** known to the classifier should be shown. When viewing only a subset of the classes one may miss some important information. For instance, if two classes can not be distinguished well by the classifier, and only one of them is shown, the user will not be able to assess this information.
5. Visualizations of different classifiers should be **comparable**. This is desirable in order to allow a straight-forward visual comparison of two or more classifiers by only learning one visualization scheme.
6. The underlying **original data** should be accessible, because this is the data users can understand and reason about. Feature representations may be too abstract in this regard, e.g. [Term Frequency - Inverse Document Frequency \(TF-IDF\)](#)-weighted features for a text document are not understandable in a straight-forward way.
7. The visualization should give an overview of the classifier's **performance** to let users get the gist of whether the classifier performs well or bad. This general information is usually the first information needed when assessing a classifier.

3 Theory

8. Because outliers may indicate errors in the classification pipeline (e.g. in pre-processing), the visualization should allow **detection** of special data items like **outliers**. Further, outliers may also influence the classifiers performance, that's why their identification is desirable.
9. The visualization should be **interactive** to allow the user to adapt the classification model through the visualization.

3.1.3 State-of-the-Art

Table 3.2 on page 48 ff. summarizes existing classifier visualizations. From the listed 32 visualizations, 22 are designed for specific classifiers, i.e. they use the structure of the classifiers model for visualization. 10 of the 32 visualization are classifier-agnostic and can be applied to almost any classifier. Among these 10, the 5 visualizations [Dol07, Rd00, FH03, PES+06, PF97] depicted in figure 3.3 are for binary classification tasks only and thus not suitable for this thesis, which deals with multi-class classification problems. The remaining 5 classifier-agnostic visualizations support multi-class classification problems. Examples of these visualizations are shown in figure 3.4 and will be discussed in the following.

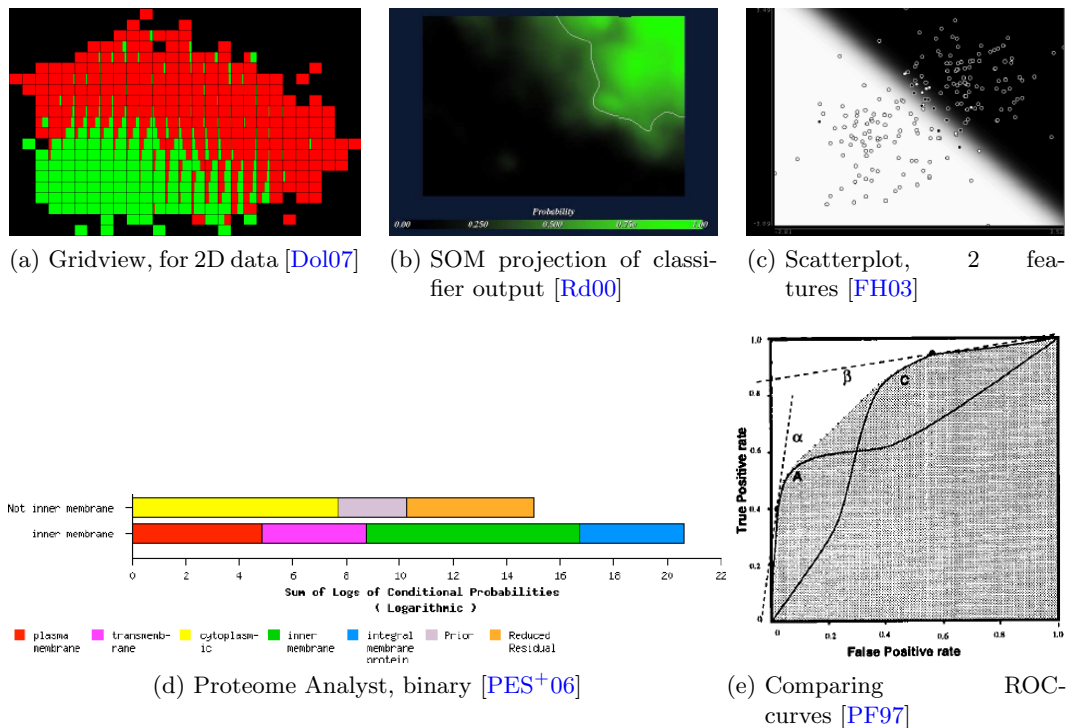


Figure 3.3: Example of *classifier-agnostic* visualizations: For *binary* classification problems.

3.1 Towards Visualizations for Arbitrary Classifiers

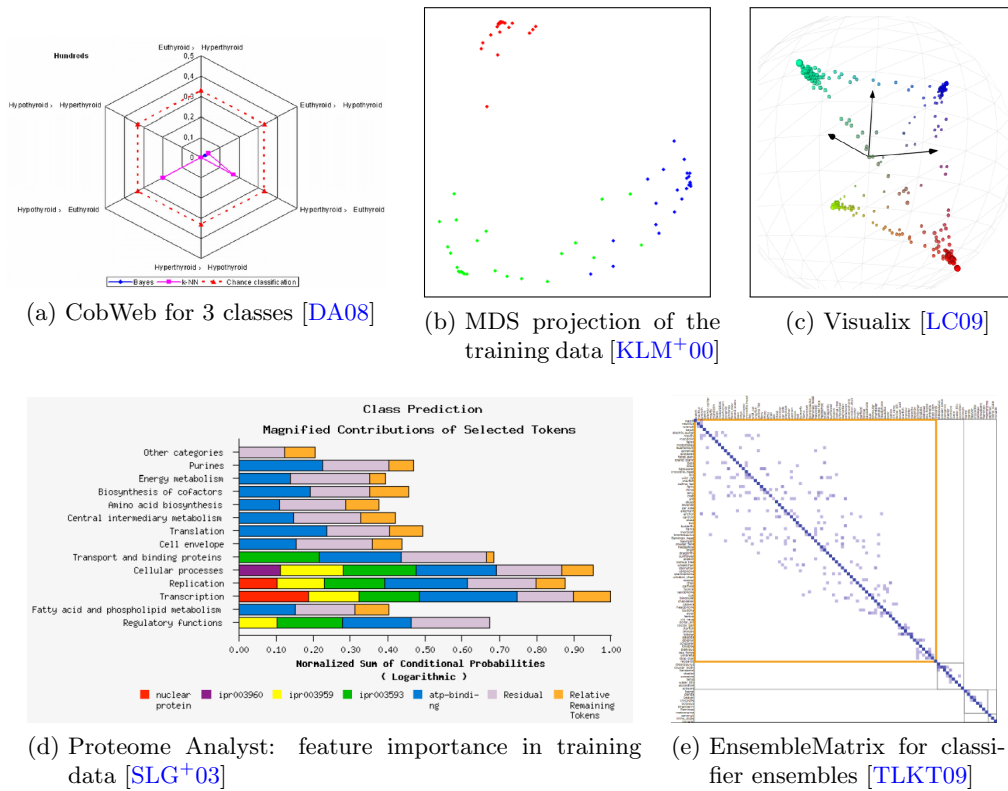


Figure 3.4: Example of *classifier-agnostic* visualizations: For *multi-class* classification problems. (a), (b), and (d) *non-interactive*, (c) and (e) *interactive* visualizations.

The CobWeb visualization [DA08] (see figure 3.4a) is based on the class confusion matrix showing how often one class is confused with another. The proposed CobWeb visualization is only suitable for a very small number of classes; in the paper examples for three classes are shown. This is because the regular polygon used for the CobWeb visualization consists of $c \cdot (c - 1)$ edges ($c =$ number of classes). Moreover, the visualization is non-interactive and does not allow feedback to the underlying classification model.

Kontkanen et.al [KLM+00] (see figure 3.4b) propose a **Multi-Dimensional Scaling (MDS)**-based projection of the data using the output of a trained Bayesian Network as similarity measure. Thus, data items classified similarly by the Bayesian Network will be placed near each other in the visualization. The projection reflects the trained model of the Bayesian network. The visualizations can be applied to any classifier that outputs a-posteriori probabilities using a suitable distance measure (e.g. Euclidean distance) for the projection. The authors show the applicability to various data sets with 2 to 4 classes. However, this visualizations is not interactive and does not allow the user to influence the underlying classification model.

3 Theory

Proteome Analyst [SLG⁺03] (see figure 3.4d) is a classification application tailored towards the bio-informatics domain. It is targeted at expert users in genome annotations. Experts may build and apply their own classifiers, whereas building a classifier means choosing training data and classifier. No further interaction is possible. According to the authors, the visualization shows which feature contributed most evidence to the classifier. But the visualization shows the training data and the relevance of the features in the training data, but there is not guarantee that the classifier does indeed use the identified features. Thus, the Proteome Analyst does not truly visualize classification models. Further, the visualization is non-interactive, this means an already trained model can not be adapted using the visualization.

The EnsembleMatrix of Talbot et al. [TLKT09] (see figure 3.4e) visualizes the class confusion matrices for classifiers. It is not clearly stated in the paper whether the confusion matrix of the training or the test data set is represented. The authors use the confusion matrix visualization of the classifiers to construct classifier ensembles in an interactive way. With this application the performance of classifiers can be assessed, and the ensemble can be interactively rearranged, such that the combined hypothesis changes. However, there is no access to single training items, and a single classifier can not be influenced, only the ensemble can be adapted.

The Model Uncertainty Visualization (see figure 3.4c) in the Visualix application [LC09] visualizes the uncertainty of classifiers decisions in the 3D space. The visualization is based on confidence values for each class (not necessarily a-posteriori probabilities) – which does not ensure that the visualization for two different classifiers are comparable. The color of a data item is derived from the colors of the classes by linear combination using the confidence values for each class. As this gives another clue of the kind of uncertainty of the classifier (which two classes compete for the item) – the final decision is not necessarily accessible. For the Visualix system a patent application³ has been made.

The derived visualization (Class Radial Visualization) in this thesis is in its basics similar to the Model Uncertainty Visualization, and has been published at the same time (both, July 2009) without being aware of the Visualix system. Section 4.1.9 points out the differences in detail after explaining the visualization derived in this thesis.

Inherent Restrictions: All multi-class visualizations aiming at visualizing classifier decisions use coloring to indicate the classes, i.e. each class has a unique color assigned. With this color-coding of classes, the number of classes that can be unambiguously visualized is restricted by human color vision. As Healey [Hea96] shows in his experiments, humans with normal or corrected-to-normal vision can only instantly distinguish 7 colors. This means when using color-coding of classes, other means have to be designed to make the visualization unambiguous.

³The patent application can be found here: <http://www.faqs.org/patents/app/20090252404>, last accessed 2011-11-12.

3.1.4 Summary

This section discussed requirements for classifier visualizations suitable to help answering the research question of this thesis. First, the need for a classifier-agnostic visualization was argued. Then, requirements for such a visualization were identified, which are in summarized form (i) interactivity, (ii) applicability to multi-class problems, (iii) showing decisions and confidence for decisions, and (iv) allow comparative assessment of classifier performance.

Reviewing state-of-the art it was found that only one such visualization exists, Visualix, which has been developed at the same time as the visualization presented later in this thesis. Although Visualix is a 3D-visualization, the idea behind both is very similar. Both visualize all data points in a layout that visually encodes the model uncertainty. Section 4.1.9 compares both approaches in detail. This visualization concept is particularly helpful for assessing classifiers on the general level and on the item-level (see section 3.1 for details of these levels).

Further, a visualization based on the confusion matrix seems suitable for assessing classifiers on the class-level, similar to the EnsembleMatrix [TLKT09] described above, but used for both, training and test data.

Thus, in this thesis two visualizations for assessing classifiers were implemented, the interactive Class Radial Visualization (see section 4.1) based on the model uncertainty for data items and the Confusion Maps (see section 4.2) based on the class confusion matrix.

Table 3.2: Overview of existing classifier visualizations, abbreviations: mc – multi-class, sl – single-label, I – interactive, FB – feedback to model

Authors	Classifier	Type	I	FB	dataset, application	comment
Becker et al., 1997 [BKS97]	Naive Bayes	mc sl	✓	-	iris, census	
Provost & Fawcett, 1997 [PF97]	all	binary	-	-		based on ROC curve
Becker, 1998 [Bec98]	Decision Table	mc sl	✓	-	census, mushroom	
Ankerst et al., 1999 [AEEK99]	Decision Tree	mc sl	✓	✓	shuttle	no classifier visualization, only data vis
Ankerst et al., 2000 [AEK00]	Decision Tree	mc sl	✓	✓	satimage, segment, shuttle, australia, DNA	extension of [AEEK99]
Kontkanen et al., 2000 [KLM ⁺ 00]	probabilistic ⁴	mc sl	-	-	iris, mushroom, ionosphere, liver disorder, mole fever	shown only data sets with 2-4 classes
Rheingans & desJardins, 2000 [Rd00]	Classifiers with probabilistic output	binary	✓	-	Adult	
Ware et al., 2001 [WEH ⁺ 01]	Decision Tree	mc sl	✓	✓	waveform, shuttle, segment, sat, letter	extension of [AEEK99]
Caragea et al., 2001 [CCH01]	SVM	mc sl	✓	-	olive oil	
Hadjarian et al., 2001 [HBP01]	Ripper, Rocchio	mc sl	-	-	reuters-21578	text classification only
Melnik & Pollack, 2002 [MP02]	feed-forward NN	binary	-	-	vowel	

⁴shown for Bayesian network

Table 3.2: Overview of existing classifier visualizations, abbreviations: mc – multi-class, sl – single-label, I – interactive, FB – feedback to model (contd.)

Authors	Classifier	Type	I	FB	dataset, application	comment
Urbanek, 2002 [Urb02]	Forrests of Decision Trees	mc sl	-	-		
Frank & Hall, 2003 [FH03]	probabilistic	binary	-	-	diabetes, iris	
Kopanakis & Theodoulidis, 2003 [KT03]	association rules	✓	-	-	only for low-dim data	
Szafron et al., 2003 [SLG ⁺ 03]	Classifiers with probabilistic output ⁵	mc sl	-	-	yeast, proteome analysis	explanation of classifier is training set, features from the data need not necessarily be used by the classifier
Cook et al., 2004 [CCH04]	SVM	mc sl	✓	-	olive oil	extension of [CCH01]
Poulet, 2004 [Pou04]	SVM	mc sl	✓	✓	aml-all leukemia, breast cancer, colon tumor, lung cancer, ovarian cancer	
Axelsson, 2004 [Axe04]	Naive Bayes	binary	✓	✓	intrusion detection	
Groth, 2006 [Gro06]	Naive Bayes	binary	-	-	discrete attributes only	
Plaisant et al., 2006 [PRY ⁺ 06]	Naive Bayes	binary	✓	✓	text classification	

⁵applied Naive Bayes, Neural Nets, **SVM**

Table 3.2: Overview of existing classifier visualizations, abbreviations: mc – multi-class, sl – single-label, I – interactive, FB – feedback to model (contd.)

Authors	Classifier	Type	I	FB	dataset, application	comment
Poulin et al., 2006 [PES+06]	additive classifiers ⁶	binary	✓	-	proteome analysis	extension of [SLG+03]
Stiglic et al., 2006 [SMPK06]	ensembles of 3 decision trees	binary	✓	✓	aml-all leukemia, breast cancer, lung cancer, prostate cancer	
Dolfing 2007 [Dol07]	all	binary	-	-	letter recognition	classification only small part of master thesis
Poulet, 2008 [Pou08]	SVM , Decision Tree	mc sl	✓	✓	aml-all leukemia, breast cancer, colon tumor, lung cancer, ovarian cancer	extension of [Pou04]
Chodos & Zaiane, 2008 [CZ08]	Associative Classifiers	binary	✓	✓	mushroom, car evaluation, nursery	
May & Kohlhammer, 2008 [MK08a, MK08b]	Decision Trees	binary	✓	✓		
Caragea et al., 2008 [CCWH08]	SVM	mc sl	✓	-	olive oil	two very similar publications extension of [CCH01, CCH04]
Diri & Albayrak, 2008 [DA08]	all	mc sl	-	-	medical data	complexity of visualization is (c), c = number of classes
Fuchs et. al. 2009 [FWG09]	Genetic Algorithms for CNF-rules ⁷	binary	✓	✓	fluid classification	

⁶Naive Bayes, linear **SVM** and linear regression, visualization depends on classifier

⁷Conjunctive normal form

Table 3.2: Overview of existing classifier visualizations, abbreviations: mc – multi-class, sl – single-label, I – interactive, FB – feedback to model (contd.)

Authors	Classifier	Type	I	FB	dataset, application	comment
Lecerf & Childlovsky [LC09]	probabilistic classifier	mc sl	✓	✓		
Talbot et al. 2009 [TLKT09]	all	mc sl	✓	✓		based on confusion matrix

3.2 Towards Summarized, Visual Representations of Text

The generation of training data for text classification is usually done manually by domain experts. This means, the texts are presented to domain experts, who manually assign class labels for each text – a repetitive, time consuming work. Approaches to reduce the overall labeling time can be grouped into approaches to reduce the amount of necessary training data and approaches to reduce the time for labeling a single text document. The former include active learning [Set10] and semi-supervised learning strategies [Zhu08]; an example of the latter is the “labeled feature approach” [DMM08]. In this section we aim at reducing the time required to label a single text document. Note, that this approach can then be combined with e.g., active learning, which means in total the time per text document as well as the total amount of texts will be reduced.

The goal is to find a condensed representation of (potentially large) text documents that would be faster processable by human labelers. While the text comprehension time should be reduced, it is crucial that the information important for labeling of the texts still remains in the alternative representation. In other words, the goal is to reduce the time humans require to make a decision about the correct label of the text, i.e. to minimize $t_1 - t_0$ in figure 3.5.

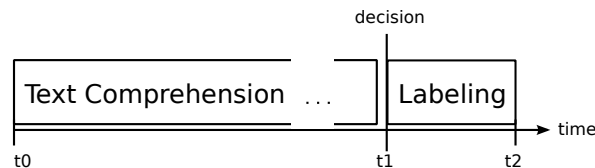


Figure 3.5: Overview of the labeling process for text classification. The time needed to comprehend texts by far extends the time for the actual labeling.

The assumption is that the information the user needs to identify the category is hidden in some key parts of the document. Conversely most parts of the document can be considered as irrelevant, they do either not contribute information for the task of finding the correct category or even distract the user from identifying the correct category. Especially for long documents, the cognitive effort for filtering irrelevant information is high. The approach of reducing time for human text comprehension by using alternative representations is only beneficial for the overall training data generation process if the alternative representations can be derived automatically.

In the next section we review state-of-the art of automatic text summarization and text visualizations methods and finally discuss their suitability for the task of generating training data for text classification.

3.2.1 State-of-the-Art

In this section we review the state-of-the art in text summarization and visual text presentation.

Text Summarization aims at producing a shorter version of the text while retaining the overall meaning and information content. Gupta and Lehal [GL10] present a review for extractive summaries from texts. Extractive summaries are a selection of meaningful document parts, while abstractive summaries are shorter re-phrasings of the text. We chose to use the TextRank algorithm [MT04] as it allows for text summarization on two different levels of granularity by extracting (i) key sentences and (ii) key phrases.

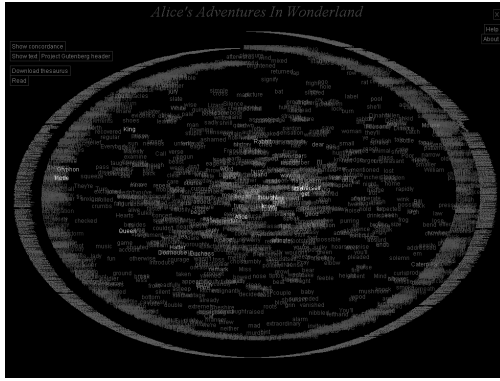
Also the field of Information Visualization has to offer ideas on alternative **text representations** [uB10]. Some examples are shown in figure 3.6. Most of the visualizations show additional aspects of the text which are not instantly accessible in full-text representations. The Word Tree [WV08] for example, is an application of a keyword-in-context method and visualizes word concordances. In TextArc [Pal02] word frequencies and distributions of all words in the text are visualized. These visualizations allow to interactively investigate and explore the texts, but are neither condensing the text nor designed as topical summaries. PhraseNet [vHWV09] shows inter-word relations and may be considered as a condensed visualization of a text as two occurrences of the same phrase are collapsed into one node in the graph. True visual text summarizations are word clouds (see section 3.3.1, such as Wordle [VWF09], or the Document Cards visualization [SOR⁺09]). The latter one also resembles a normal word cloud in absence of tables or images in the documents and uses the layout algorithm presented in this thesis (and published in [SKK⁺08]).

Regarding our task of identifying the label for a given text as fast as possible, a clear and simple layout seems most beneficial. Visualizing non obvious text features like word-concordances and contexts of keywords is not necessary and could easily distract the user. Thus, word clouds seem a suitable visualization for this task.

3.2.2 Summary

Summing up, existing text summarization methods allow for summarizing text on different levels of granularity: (i) key sentences, and (ii) key phrases. It is not clear which of both would be more suitable for the task of labeling documents. Therefore, we will perform a comparative user evaluation to assess their suitability. Regarding the layout, we will use normal line-by-line layout for key sentences. Presenting key phrases as word clouds seems to be straight-forward. Using the word cloud representation allows for an additional encoding of the importance of each key phrase, which is usually done by varying the font size in the word cloud.

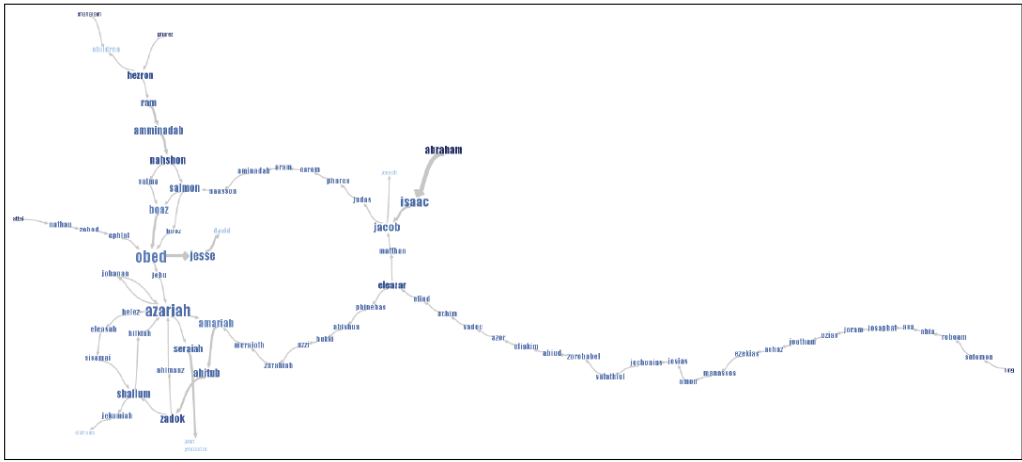
3 Theory



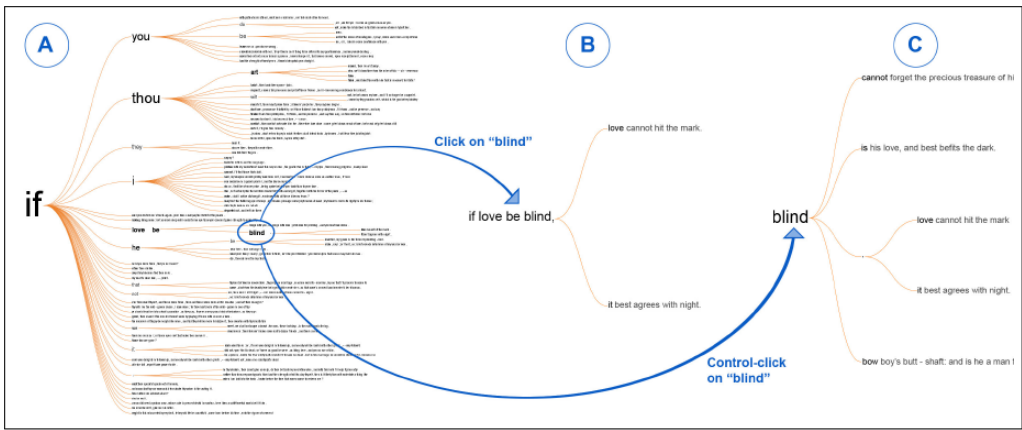
(a) TextArc [Pal02]



(b) Wordle [VWF09]



(c) PhraseNet [vHWV09]



(d) Word Tree [WV08]

Figure 3.6: Examples of visual text representations. Only the Wordle word cloud can be regarded as true text summarization. All others visualized aspects of the text than is instantly accessible in the full-text.

3.3 Towards Visualizations for Text Classification

This section discusses possibilities of visualizing the feature level. As argued in the introduction of this chapter (on page 37) we focus on the task of text classification. This means, the original data to classify are text documents. Using a vector-space model representation, simply speaking each feature dimension corresponds to a word.⁸

In table 3.1 the following aspects of the feature level were identified:

1. Relevance of features for decision
2. Ambiguity of features
3. Dependency of features

For texts this can be specified to

1. Relevance of single words for decision
2. Polysemy and synonymy of words
3. Dependency of words

Clearly, the first aspect (relevance of words for decision) is the most important one. Polysemy and synonymy of words as well as dependency of words would not give any insight in the classification model if these words were actually never used for making predictions. Further, humans are better than any automatic method to detect polysemy and synonymy of words. Natural-language texts inherently consists of many dependent features, which can also be exploited for text classification, see for example [NSC06]. Because feature ambiguity and dependency is more the rule than the exception for text documents in this work we focus on visualizing the relevance of single words for the decision.

In order to visualize the relevance of features for a decision it is necessary to represent (i) the decision, i.e. the class and (ii) the features for each class. Additionally, it would be helpful to visualize the similarity of the classes on the basis of the features they were trained on. In the previous section (see page 52), state-of-the art for text visualization has been reviewed already. For the task of visualizing the feature level, only a visualization similar to tag clouds seems reasonable. This is because: (i) In classification tasks, usually many features are present, thus a space-filling visualization is desirable. (ii) In the vector-space model no inter-dependence between features is represented, thus visualizations showing feature relations are not applicable. Thus, we sketch the visualization as follows:

- Use an area to represent a class.
- Place the relevant key words for this class inside the area.
- Place the area such, that nearby areas correspond to classes similar to each other.
- Use a space-filling approach.

⁸More specifically, this representation depends on the creating of the vectors space, which is explained in detail in section 2.3.

The conditions (i) area for a class, and (ii) space-filling approach point towards the application of Voronoi diagrams to create the area subdivision of a 2D space. Voronoi diagrams are briefly introduced in section 2.6. A Voronoi subdivision of a 2D space generates convex polygons. Therefore we need an algorithm that can place key words (or simply words) in arbitrary, convex polygons, preferably in a space-filling way. The next section reviews state-of-the art of such algorithms – mostly used in context of tag clouds.

3.3.1 Tag Layout

The standard tag cloud layout used by Web 2.0 websites such as del.icio.us and flickr uses a rectangular line-by-line layout with different font sizes to indicate the relevance of the tags. The tags are sorted by their relevance or alphabetically. Kuo et. al [KHGW07] further include color for the tags to indicate the recency of the tag, which is calculated from the average publication data of the underlying documents (see figure 3.7c). These visualizations naturally produce a lot of white space, an issue which was tackled by Kaser & Lemire [KL07] who used an [Electronic Design Automation \(EDA\)](#) packing algorithm to layout tags in nested tables for HTML based websites (see figure 3.7e). Shaw [Sha05] was the first who broke-up with the rectangular layout and further introduced the visualization of relations between tags. His tag map is a graph-like 2D representation of tags, where tags correspond to nodes and related tags are connected by an edge (see figure 3.7b). However, the visualization contains a lot of white space and tags are overlapping each other, problems that were also not solved by Stefaner [Ste07] who introduced elastic tag maps. In the elastic tag maps visualization the tags are placed in an nearly circular 2D space using [Principal Component Analysis \(PCA\)](#) and [Curvilinear Component Analysis \(CCA\)](#), with the most relevant tags defining the extremes of the spanned space (see figure 3.7d). Bielenberg and Zacher [BZ05] also proposed a circular layout, where the font size and position from center to orbits show the relevance of the tags. The most relevant tags are placed in the center of the circle and referred to as focus. Also in this visualizations huge white space areas and overlaid tags remain (see figure 3.7a).

Note, that the the tag layout algorithm presented in this thesis was published in 2008 in [SKK⁺08], i.e. before the following papers.

Wordle, a web-based application [VWF09] creates space-efficient word clouds but aligns some of the words vertically (see figure 3.7f), which makes them hard to read. This issue is tackled by the word bridge tag layout [KKEE11]. In the word bridge layout, tag cloud serves as nodes and links in a node-link diagram, therefore space-efficiency and readability of single tag clouds is of high interest. The tag layout of word bridge is inspired by Wordle but positions all words horizontally (see figure 3.7g and 3.7h). A tag cloud preserving semantic relations between tags has recently been proposed by Wu et al. [WPL⁺11]. The semantic tag cloud is constructed in three steps. First, extracted keywords are placed in a scatter-plot layout. Then overlap is reduced. Second, irrelevant

words are removed and the resulting rectangular white space areas (called seams) are cut resulting in a compact layout. Third, bubble sets are added to enclose related tags. An example of the semantic word cloud is shown in figure 3.7i. Schrammel et al. [SLT09] investigated the influence on different tag orderings in a user study. User’s tasks in this study were finding specific tags, finding tags related to a topic and recalling tags. The authors found out, that there is no single optimal layout, but the best layout depends on the task at hand.

All presented algorithms can either not compute a space-filling layout inside a restricted boundary or can only handle rectangular boundaries. The algorithm developed in this thesis (section 4.3) and published in [SKK⁺08] is the only one that is able to lay out tags compactly inside an arbitrary, convex polygon.

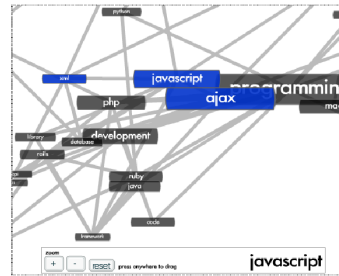
3.3.2 Summary

In this section a visualization for text classification models on the feature level was sketched. This visualization is able to represent the relevance of features for the final decision (the class). We described desired properties of such a visualization and found that a combination of Voronoi diagrams and tag layout in arbitrary, convex polygons fulfills these properties. Review of existing tag layout literature indicated that no such tag layout algorithm exists. Therefore, we designed such an algorithm (see section 4.3). The implementation of the sketched visualization is described in section 4.4.

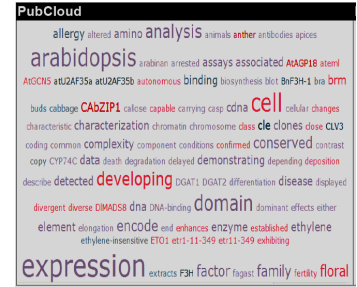
3 Theory



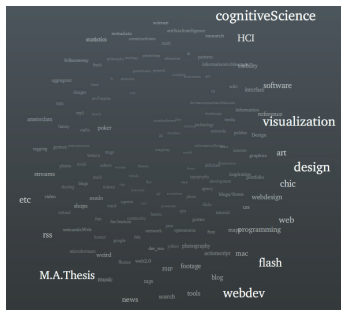
(a) Circular layout [BZ05]



(b) Graph layout [Sha05]



(c) PubCloud layout [KHGW07]



(d) Elastic tag maps [Ste07]



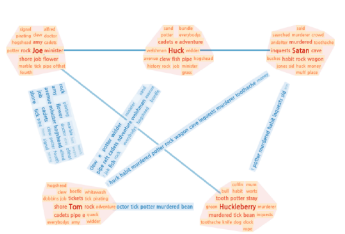
(e) Nested HTML tables [KL07]



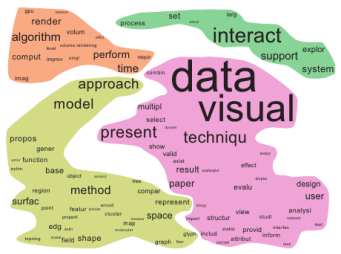
(f) Nested HTML tables [VWF09]



(g) Word bridges layout [KKEE11]



(h) Word bridges layout example [KKEE11]



(i) Semantic tag clouds [WPL+11]

Figure 3.7: Tag cloud examples sorted by year of publication. (a) and (b) 2005, (c) to (e) 2007, (f) 2009, (g) to (i) 2011

3.4 Towards Visual Active Learning

In this section the concept of Visual Active Learning is introduced. Visual Active Learning can be seen as a modification to active learning giving more power to the user. This concept has first been published in [SG10].

In short, in classical active learning an algorithm selects new items to label based on the classifier's decisions and statistics on the unlabeled data set. The algorithm aims at selecting the presumably most informative items, i.e. the items that – if the label were known – would increase the classifier's performance most. (see section 2.5 for details).

In Visual Active Learning, not the algorithm selects the items to label, but the human herself. Similarly, to the algorithm that uses different statistics to make its decision, the user bases her decision on a visualization of the current classification model. Clearly, the crucial point for successful Visual Active Learning is the visualization. The idea is that in this way the user's background knowledge can be used and indirectly integrated into the classification model.

The concept of Visual Active Learning is outlined in the following. First, state-of-the-art of active learning is briefly reviewed in section 3.4.1. The differences and similarities between active learning and Visual Active Learning are described in section 3.4.3. Requirements of the crucial ingredient of Visual Active Learning – the visualization – are listed in section 3.4.4.

3.4.1 Active Learning

This section briefly reviews state-of-the-art literature for active learning focusing on classifier-agnostic active learning strategies. Detailed description of the Active Learning scenario can be found in the foundations section 2.5 on page 33.

Settles presents an up-to-date review [Set10] of active-learning literature focusing on querying strategies and reviewing experimental findings. While the active learning approach (see section 2.5 for details) seems quite natural, users have to fully trust the active learning strategy which is not always the case [TO09]. Interestingly, depending on the application scenario, some active learning strategies require more labeled instances than passive learning or have been outperformed by the random baseline ([SU07, Gas09, Set10, GS07, LC06]). Furthermore, Baldrige and Palmer present evidence [BP09], that the efficiency of active learning correlates with the proficiency of the annotator. Active learning strategies tailored towards a special hypothesis class, as for instance for support vector machines [SC00] are not the focus of this work.

3.4.2 Concept

The key idea of Visual Active Learning is to give the user the power and responsibility to select the items as opposed to let them be selected by an automatic procedure. The main motivation concerns a better utilization of human intelligence which goes beyond simple labeling. The idea is that the user may select examples that are important and informative for the classification task using her domain knowledge.

If the selection is supported by a visualization showing the uncertainty of the classifier's decision, and thus showing which examples would be selected by an automatic greedy active learning procedure, the user may also simply accept the examples proposed by the automatic procedure. This means, Visual Active Learning tightly integrates automatic and manual approaches letting the user make the final decision which item to select.

The general procedure of Visual Active Learning is outlined in figure 3.8. ① A classification model is built on labeled data. ② The classifier is applied on the unlabeled data. ③ The classification results on the unlabeled data are visualized. Either the visualization is specifically designed for the used classifier or a classifier-agnostic visualization is used. ④ The user selects the next items to label in the visualization based on her knowledge and the patterns in the visualization. ⑤ The human labeler assigns the label. ⑥ The newly labeled instance serves as additional training data for the classifier. The process continues at step ① with the extended labeled data set.

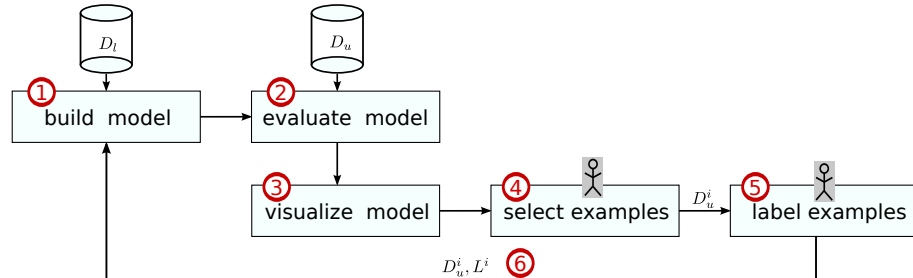


Figure 3.8: The Visual Active Learning scenario. ① A classification model is built on labeled data. ② The classifier is evaluated on the unlabeled data. ③ The classifier's results on the unlabeled data are visualized. ④ The user selects the next items to label. ⑤ A human labeler assigns the labels. ⑥ The newly labeled instances serve as training data for the new classifier. The process continues in a loop.

Obviously, the visualization is the crucial point for the successful selection of unlabeled data items by the user. Requirements for such a visualization are described in detail in section 3.4.4.

3.4.3 Comparing Active Learning and Visual Active Learning

In table 3.3 active learning and Visual Active Learning are compared. In Active Learning, an algorithm selects the example and the human labels it. In Visual Active learning, humans do both, selecting and labeling. In order for the human to be able to reasonably select examples, the classifier’s results on the unlabeled data is visualized. Both, Active Learning and Visual Active Learning can be performed classifier-dependent and classifier-independent by the appropriate choice of the selection strategy or visualization. Active Learning degrades the user to a pure labeling machine, while Visual Active Learning allows her to select and label examples herself. Conceptually, the key motivation of Active Learning is, that the automatic selection algorithm knows best which examples would be best suited to improve the classifier’s performance. However, experiments show, that this assumption is not valid in general [Set10]. The key motivation of Visual Active Learning is, to find a way of exploiting the annotators domain knowledge beyond the simple labeling of data items.

Table 3.3: Comparison of Active Learning and Visual Active Learning.

	Active Learning	Visual Active Learning
item selector	algorithm	human
basis for selection decision	classifier results and statistics on unlabeled data	visualization of classifier results on unlabeled data
item labeler	human	human
classifier types	depending on selection strategy, classifier-independent and dependent available	depending on visualization

3.4.4 Requirements for a Visualization for Visual Active Learning

A visualization supporting Visual Active Learning should retain four crucial properties. First, the visualization should allow to judge problematic behavior of classification models like for example biases towards particular classes. Second, fast and easy identification of false and/or problematic examples, e.g. outliers should be supported. Third, users should be able to rapidly select and label batches of examples, not only single examples. Fourth, the visualization should be classifier-independent and usable on top of any multi-class, single-label classification task which provides a-posteriori probabilities. This requirement for classifier-independence comes from practical reasons. As outlined in a recent survey on the use of active learning strategies in the natural language processing domain [TO09], practitioners highly appreciate classifier-independent strategies.

3 Theory

In general, Visual Active Learning could not only be applied to single-label, but also multi-label classification problems. However, the latter are beyond the focus of this thesis.

3.4.5 Summary

In this section the concept of Visual Active Learning was presented. In Visual Active Learning not the algorithm selects the items to label – as in classical Active Learning – but the human herself. Similarly, to the algorithm that uses different statistics to make its decision, the user bases her decision on a visualization of the current classification model. Clearly, the crucial point for successful Visual Active Learning is the visualization. The Class Radial Visualization is suitable for Visual Active Learning and used in the experiments in section [5.3](#).

“Data! Data! Data!” he cried impatiently. “I can't make bricks without clay.”

(*Sherlock Holmes in The Adventure of the Copper Beeches*)

4 Implementation

In this chapter four visualization modules are presented which have been developed to answer the research questions of this thesis. Table 4.1 gives an overview of the modules.

Table 4.1: Overview of all developed visualization presented in this chapter.

Visualization	Represented Data	Aspect
Class Radial Vis →Sec 4.1	classification results, classifier quality, underlying data	Assessment, Understanding, Adaptation
Confusion Maps →Sec 4.2	class confusion matrix	Assessment
Tag Clouds →Sec 4.3	key words or tags of document or document collection	indirectly Construction, see experiment in Section 5.4, indirectly Understanding (used for Voronoi Word Clouds)
Voronoi Word Clouds →Sec 4.4	details of document classes learned from examples	Understanding

The *Class Radial Visualization* is designed to visually assess and understand decisions and results of a classifier. It is based on a-posteriori probabilities and can therefore be applied to any classifier that outputs or whose output can be mapped to a-posteriori probabilities. A simple-to-use interaction mechanism allows users to adapt the classification model.

Confusion Maps are a visual representation of class confusion matrices, allowing faster assessment and detection of patterns in the class confusion matrix. Further, aspects of classifiers, like over-fitting, can be visually identified by comparing Confusion Maps of training and evaluation sets.

4 Implementation

The family of *Tag Layout* algorithms allows to place tags in arbitrary, convex polygons. The special tag layout is a prerequisite for the Voronoi Word Cloud and used in the experiments for classifier construction in section [5.4](#).

Voronoi Word Clouds combine Voronoi subdivision and the tag layout for arbitrary convex polygons mentioned above. Voronoi Word Clouds can be applied to visualize the model of a specific text classifier as shown in section [5.6](#) on page [155](#).

4.1 Class Radial Visualization

The Class Radial Visualization is designed to visually access the decisions and the quality of a trained classifier. Further, it allows to adapt the classifier by simple drag and drop interaction. Basically the visualization shows representations of all classes and all items in the classified data set. The item's color indicates the classifier's decision and the items position indicates the confidence of the decision. The visualization allows to find items that are probably misclassified and further, to correct the decision of the classifier. The visualization was first published in [SL09a] and the idea of the feedback mechanism was published in [SL09b].

4.1.1 Method

In the Class Radial Visualization all class items are displayed equally distributed around the perimeter of a circle. Each class item is represented by a square and its border is drawn in the unique class color. Initially, all squares are unfilled. Then, the square of the class with the maximum number of assigned test items is completely filled with its class color. The squares of all other classes are filled with their colors proportionally to the number of assigned items. The class items are placed according to the order of the classes in the input set. The trained classifier is used to calculate the a-posteriori probability distribution for each test item. Each test item is initially positioned in the center of the circle. The direction vectors from the test item to all class items are computed. The initial length of these vectors is set to the radius of the circle. Then, each direction vector is weighted by the according class probability. The vector sum of the weighted direction vectors yields the final position of the test item. Finally, a test item is colored according to the class with the highest confidence.

Algorithm 1 summarizes the layout step (without the coloring of the items and classes). Intuitively, one can think of springs attached between items and classes. Each spring attracts the item towards the position of the class in the plane with a force proportional to the a-posteriori probability $p(y|x)$ of assigning item x to class y . The final position of the item then corresponds to the equilibrium state where all spring forces sum up to zero.

For instance, an item with probability distribution $p = (p_1, \dots, p_c)$, $p_i = 1$, $p_j = 0 \forall j \neq i$ lies exactly on the location of the class item associated with class i . In this case the classifier is 100% sure that the test item belongs to class i . On the contrary, a test item whose a-posteriori probability corresponds to a uniform distribution is placed in the center of the circle. This indicates that the classifier can not confidently assign the item to one class. The color of the item is defined by the class with maximum confidence. If the probability distribution does not have a unique maximum, the class of the first maximum found defines the color of the item.

Yet, the Class Radial Visualization can be ambiguous due to a dimensionality problem.

4 Implementation

Require: classifier, set of classes C , data set D

```

 $\alpha \leftarrow 0;$ 
 $\Delta\alpha \leftarrow \frac{2\pi}{|C|};$ 
for all  $c \in C$  do
     $\vec{p}_c \leftarrow (\cos(\alpha), \sin(\alpha));$ 
     $\alpha \leftarrow \alpha + \Delta\alpha;$ 
end for
for all  $x \in D$  do
     $\{(c, v)\} \leftarrow \text{classify}(x), v \dots$  confidence for class  $c;$ 
     $\vec{p}_x \leftarrow (0, 0)^T;$ 
    for all  $(c, v) \in \{(c, v)\}$  do
         $\vec{d}_c \leftarrow v \cdot \vec{p}_c;$ 
         $\vec{p}_x \leftarrow \vec{p}_x + \vec{d}_c;$ 
    end for
end for
return positions of examples ( $p_x$ ) and classes ( $p_c$ ) in the unit square

```

Algorithm 1: Layout algorithm for the Class Radial Visualization

The visualization space is two-dimensional in contrast to the dimensionality of the probability distribution space that depends on the number of classes c . Clearly, for $c > 2$ the mapping from the probability distribution to the two-dimensional visualization space can not be an injective function. More specifically, different probability distributions can map to the same position in the visualization space. This ambiguity is illustrated in the following examples: Figure 4.1 depicts six example distributions for $c = 4$ and $c = 5$. The x-axes show the class labels and the y-axes the corresponding classifier confidences. In figure 4.2a the layout for the examples in the 4-class case is shown. It can be seen that Example 2 ($p = (0.5, 0, 0.5, 0)$) and Example 5 ($p = (0.25, 0.25, 0.25, 0.25)$) map to the same location, the center of the circle. However, when comparing the distributions for Example 2 and 5, it is obvious that the classifier is more unsure for Example 5: Example 5 is assigned to all four classes with equal confidence, whereas Example 2 is assigned to only two classes with equal confidence. In figure 4.2b Examples 2 and 5 for $c = 5$ are shown. With five classes the visualization of Examples 2 and 5 is no longer ambiguous, because they map to different locations. Yet, other probability distributions might exist that exhibit ambiguities. Examples that can be reliably predicted by the classifier (e.g. Examples 1 and 6) are clearly mapped next to the according class item.

To sum up, the examples show that the visualization enables the user to distinguish between items that could be clearly classified and items with no confident prediction. Yet, in the latter case, the visualization does not reflect all degrees of the classifier's uncertainty. Hoffman [Hof99] also reports the ambiguity problem and proposed several variants of the RadViz visualization, which, for a given dataset, could reduce the overlapping problem. However, the general problem of mapping a higher dimensional space to a lower-dimensional still remains.

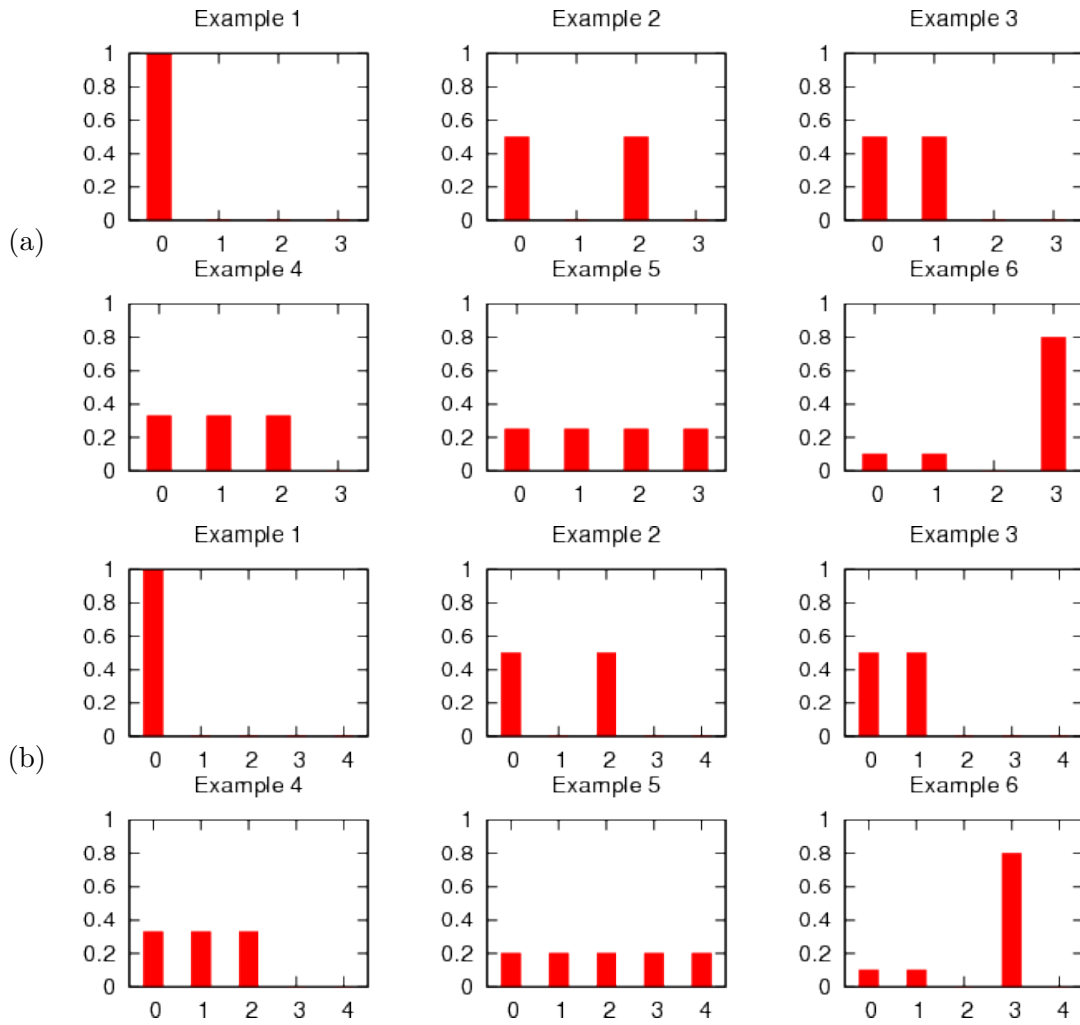


Figure 4.1: Histograms of different examples in the 4 and 5-class case.

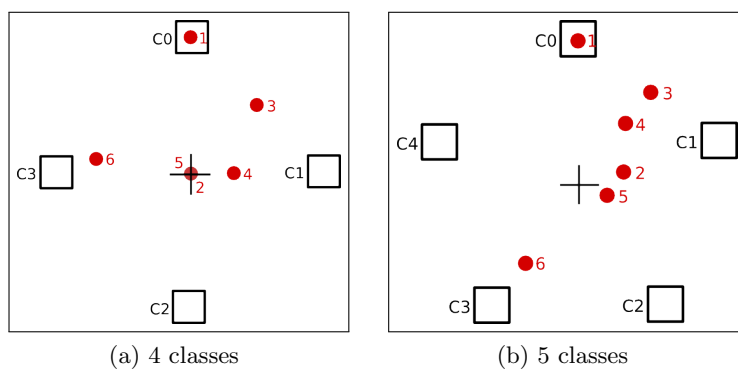


Figure 4.2: Layout for the examples histograms in figure 4.1

4 Implementation

The problem of ambiguity is resolved by adding interaction mechanisms. Additional information is shown when the user moves the mouse over a test item. Lines are drawn from the test item to all class items, while line thickness indicates the confidence for the respective class. A line to a class is only drawn when the confidence value is above a threshold θ to reduce clutter in the visualization (see section 4.1.4 for further discussion).

Feedback

By using drag and drop the user can move an item towards a class. While moving the item lines indicate the current target of the movement. If the mouse button is released the item is removed from view – it is not displayed anymore because it does not belong to the items with unknown target label anymore.

The actual influence of user feedback to the underlying model can be implemented in various ways. First, there can be instant feedback, i.e. after moving one item the classifier is updated. Alternatively, the labels can be collected for multiple items and pushed to the classifier at once. The former is an instance of serial-mode active learning, the latter would be batch-mode active learning. The visualization is updated once the classification model has changed, that is after one item in the serial-mode and after multiple items in batch-mode, has been moved by the user. The implemented application allows the user to chose the type of this feedback in the preferences panel.

4.1.2 Special Views

Two extension have been made to the standard visualization described above. The first, the magic lens view, tackles the problem of ambiguity. The second, the misclassification view, shows more aspects of classifier performance in terms of class confusion. Both views are described in the following.

Magic Lens

By the definition of the layout algorithm (see Algorithm 1), there might be multiple data samples on the same location in the visualization. There are two possible causes: First, the items can have the same a-posteriori probability distribution or a very similar one. Second, even items with different a-posteriori distributions can be located at the same point in the visualization, i.e. the visualization is ambiguous as described above. The solution to make such items accessible to mouse clicks, is the implementation of a magic lens [FS95]. Moving the lens over the area with multiple items overlays an rectangle at the location of the mouse pointer in the visualization. Inside this rectangle all items from inside the lens area are aligned next to each other which makes them accessible by the mouse pointer. The lens can be fixed on a spot, thus making the drag-and-drop style feedback possible. Figure 4.3 shows an example of the magic lens view.

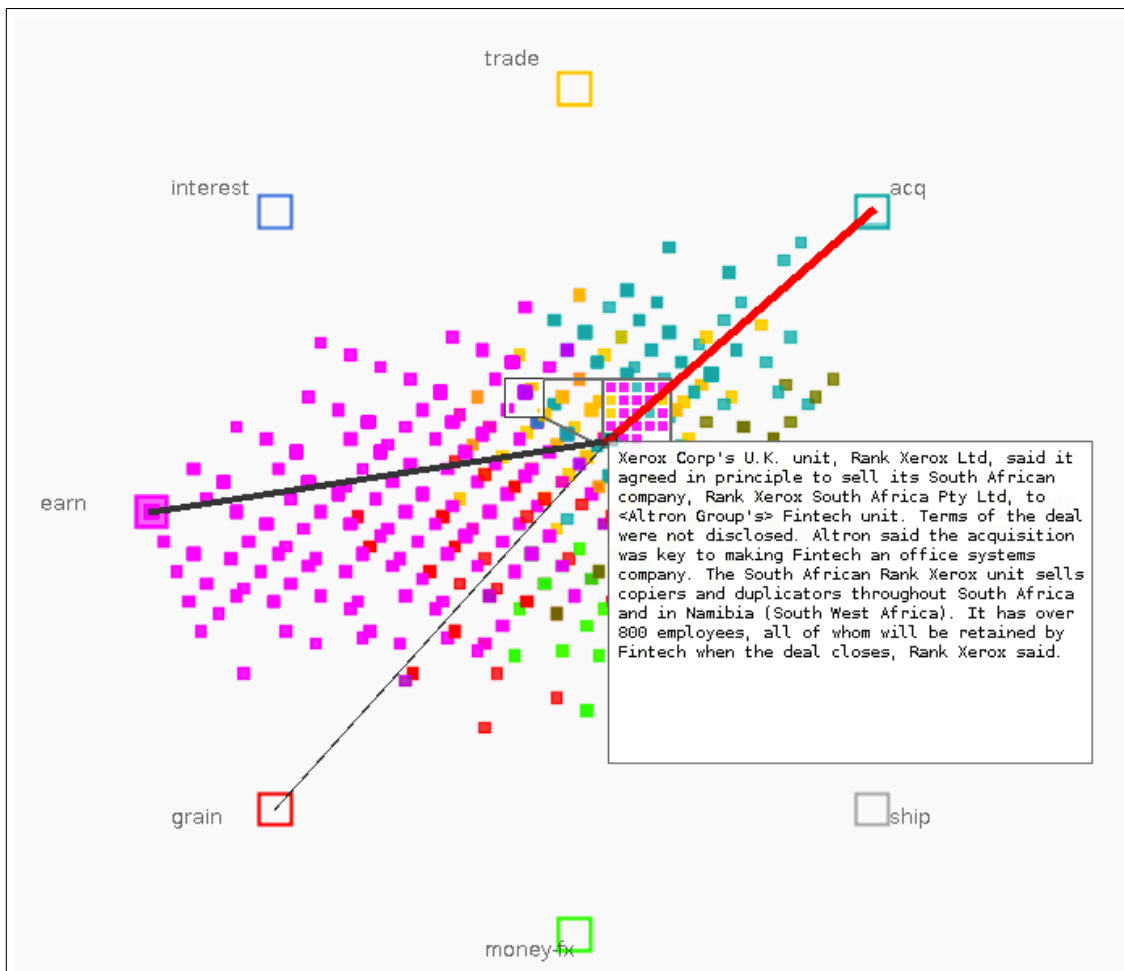


Figure 4.3: Magic lens view of the Class Radial Visualization. The user can change the (pixel) size of the lens, move over the visualization and fix the lens on an area of interest. The content of the lens (left, smaller rectangle) is displayed in the larger rectangle without overlap. Now, previously not selectable items become selectable.

Misclassification View

For assessing the performance of a classifier it is not only important how many errors it made, but also which types of errors were made. For instance, class confusion matrices 2.4.4 can give insights into which class could be clearly separated from another or whether any two classes are always confused. Confusion matrices are based on binary decisions, e.g. one could not say whether the correct class was missed by a large margin or only slightly. Even if the correct class was missed, there is a qualitative difference in an a-posteriori distribution which is nearly uniform and an an a-posteriori distribution with a clear peak for the wrong class.

One method to make the quantity and quality of class confusion accessible is the misclassification view. It is based on the layout algorithm shown in Algorithm 1. This means, the items' location in the normal view and the misclassification view are the same. Thus, both views are directly comparable. The only differences in the views are the symbol and color used for the item. Correctly classified items are displayed using a green cross (“+”) and incorrectly classified items are displayed using a red X (“x”).

The misclassification view can also be applied to only one class, which will display all items correctly assigned to this class with a green cross (“+”), all misclassifications for this class with a red X (“x”). All other items are marked with a gray circle (“o”). Note that, in order to use this view it is necessary that the data set contains the ground truth labels. Figure 4.4 shows the misclassification view for all classes (left) and a single class (right).

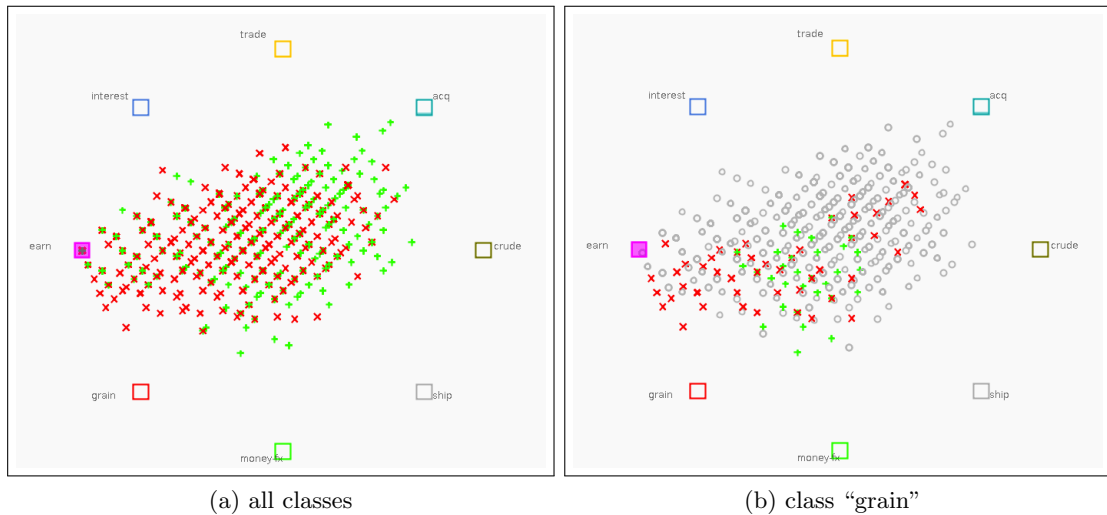


Figure 4.4: Misclassification view showing the distribution of correctly (green “+”) and incorrectly classified items (red “x”). (a) misclassification for all classes, and (b) misclassification for class “grain”.

4.1.3 Embedding in an Application

The Class Radial Visualization is the main part of the application depicted in figure 4.5. The application is briefly described here to show the idea how the visualization components may be combined.

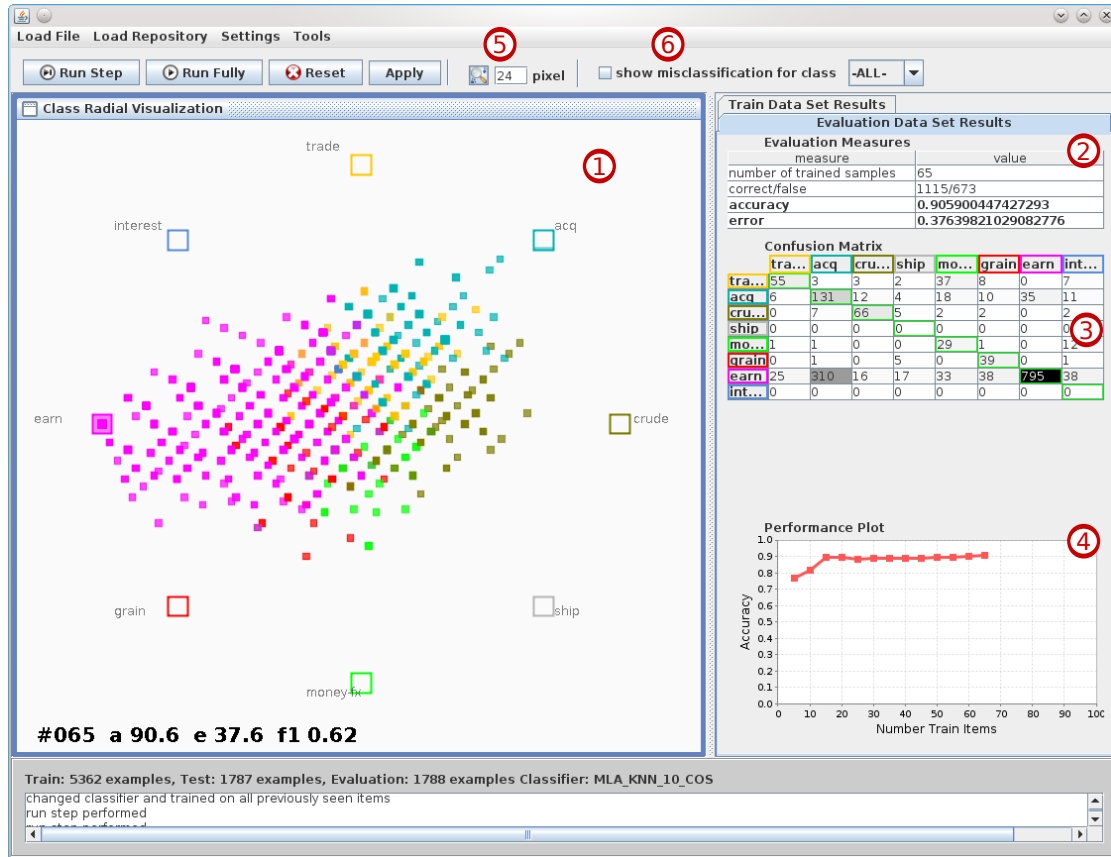


Figure 4.5: The Class Radial Visualization application consists of: ① the Class Radial Visualization, ② a table of performance measures, ③ the class confusion matrix, ④ a performance plot showing the performance of the classifier depending on the number of training items, ⑤ controls for the magic lens and ⑥ the misclassification view (the latter two are described in section 4.1.2).

The top menu contains data management tools (load, save data sets) and settings (change classifier, toggle instant retraining). The “Tools” menu allows access to the class Confusion Map visualization and histogram visualization showing the distribution of items over the classes. The main frame consists of the Class Radial Visualization ① and the evaluation panel (on the right). The evaluation panel shows a table of evaluation measures ②, a colored class confusion matrix ③ and a performance plot ④. The evaluation panel can be switched to either show the performance measures on the training data set or on the evaluation data set (if available). Comparing the training data set and the

4 Implementation

evaluation data set might help to identify whether a classifier is over-fitting, i.e. if the performance measures are (nearly) perfect on the training data set and very bad on the evaluation data set. The performance plot ④ shows the accuracy of the classifier over time (i.e. over subsequent training steps). When the slope of the accuracy line is nearly zero for multiple subsequent training steps, this indicates that new training examples do not provide any more information to the classifier, i.e. the classifier does not improve any more. The button ⑤ toggles the magic lens view as described in section 4.1.2. The size of the lens can be adapted to suit the item distribution in the Class Radial Visualization ①. The the tool-bar ⑥ provides access to the misclassification view as described in section 4.1.2. For a fully labeled test data set this allows to see which items are misclassified and whether there is a certain pattern in the misclassifications. At the bottom of the application window detailed logging output is available.

4.1.4 Parameters

The parameters of the visualization are the color palette, and the icons for both, classes and items. Further, the threshold θ defines which confidence lines are drawn. An a-posteriori probability of an item for a classes below θ is not displayed by a line while hovering over the item. Throughout this thesis $\theta = 0.1$. Depending on the classification problem other values for θ may be more suitable or required by user.

4.1.5 Feedback Example

The example in figure 4.6 shows the process of the user feedback. The COIL-20 image data set (see section 5.1.2 for a description) was classified with the **k-Nearest Neighbor (KNN)** classifier. In figure 4.6a a test item obviously assigned to the wrong class is selected. The selected item is assigned to the class “cup” (indicated by the thickest a-posteriori line) whereas the test item actually belongs to the class “duck” (shown by the underlying original data). The user now can move the item towards the correct class “duck” as shown in figure 4.6b. The moved item then serves as new training example for the particular class and the classifier is re-trained and re-evaluated. The new visualization for the test items can be seen in figure 4.6c.

4.1.6 Properties

The Class Radial Visualization is suitable for classifiers that output a probability distribution over classes or whose output can be mapped to a probability distribution.

All desired properties for a classifier-agnostic visualization listed in section 3.1.2 are fulfilled. In detail, the Class Radial Visualization

1. supports multiple classes,
2. uses a-posteriori probabilities,

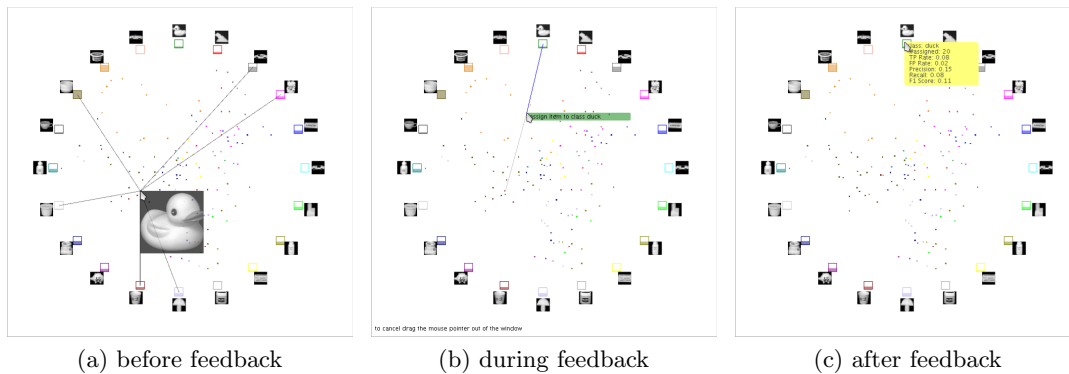


Figure 4.6: The feedback process with the Class Radial Visualization. Data items are moved by drag and drop. (a) The image of the duck is obviously misclassified. (b) The user moves the item to the correct class. (c) The classifier is retrained and has one more training example.

3. shows the final decisions and confidences,
4. shows all classes known to the classifier,
5. is comparable (two visualizations of different classifiers can be visually compared),
6. allows access to the underlying data,
7. gives an overview over the performance (and more so by embedding it in the proposed application),
8. allows to detect outliers, and
9. is interactive.

The number of classes to distinguish visually is limited by the number of colors that the human eye can distinguish. For nominal encoding – i.e. one color for each class – this is limited to 5-10 different colors [Hea96]. This means, 5-10 different colors can be distinguished immediately by the human eye. If more colors/classes are needed, the visualization becomes ambiguous and the ambiguity has to be resolved. In the interactive mode this can be done by lines connecting the items to the classes, using a highlighted line for the most probable class.

Moreover, the visualization does not reflect the similarity of classes. A similarity-based layout can be achieved as follows: (i) for each class calculate the mean vector of its training items, (ii) apply 1-dimensional **Force-Directed Placement (FDP)** using any similarity measure on the mean vectors. However, the current implementation does not use this similarity-based layout of the classes, because the similarities would change in every training step (as the mean vectors of the training data change), which would cause a rearrangement of the classes. A rearrangement of the classes would confuse the user, because the whole visual context changes (all class items and all test items).

Currently the visualization is not context preserving over time. This means when the layout changes, it changes in one step. Due to change blindness and inattentional blind-

4 Implementation

ness in human perception [CMS99] users will not be able to detect all changes in the visualization if it moves from one layout to another. A solution to this problem would be morphing the visualization and/or clearly highlighting important parts of the visualization.

The computational complexity of calculating the layout of the visualization is $O(n)$ with n being the number of items to lay out. This linear-time layout makes the visualization suitable for interactive (real-time) usage.

4.1.7 Supported Tasks

This section gives an overview of tasks supported by the visualization and shows some examples. More examples are given in the implementation section 5.2 on page 103.

Get a Gist of the Classification Model: The visualization allows to get a general overview of the classification model. The following questions can be answered:

- Q1: Are there classes that attract more items than others? See for example class “earn” in figure 4.7b (leftmost class, light blue color). It attracts nearly all items.
- Q2: Are there classes that attract no items? See for example class “crude” in figure 4.7b (rightmost class, red color). It attracts no item.
- Q3: Are there any two competing classes (items have high probability of belonging to either of them). See for example the classes “politics” and “economy” in figure 4.7a (topmost and rightmost class). Many items are positioned along the boundary between the two classes.
- Q4: Is there a general pattern how items are assigned to classes? In figure 4.7c all items are nearly symmetrically assigned to the classes, in contrast to figure 4.7b where most items are assigned to only a subset of the classes. The general pattern of the item distribution in figure 4.7c seems not very informative (many items in the center, equally distributed to the borders). It seems as if the classifier could not learn much on the data set, which turns out to be true. The features were badly engineered for the classification task (using nouns for sentiment detection), such that the classifier basically did randomly guess the class label.

Furthermore, access to the underlying data via mouse-over revealed noise in the data set, as can be seen in figure 4.7c. An item with no textual content is present in the training set. This discovery can be a hint for further investigating the correctness of the preprocessing steps (web crawling, text preprocessing).

Find Items with Low Confidence: The task to find item with low confidence is well supported in the visualization. These low-confidence items are either in the center of the circle or at the boundary lines between classes. Such items can be instantly accessed and via the mouse over function it becomes clear between which classes the item is not clearly decidable. See for instance the item in figure 4.7a. Its text is about a soccer

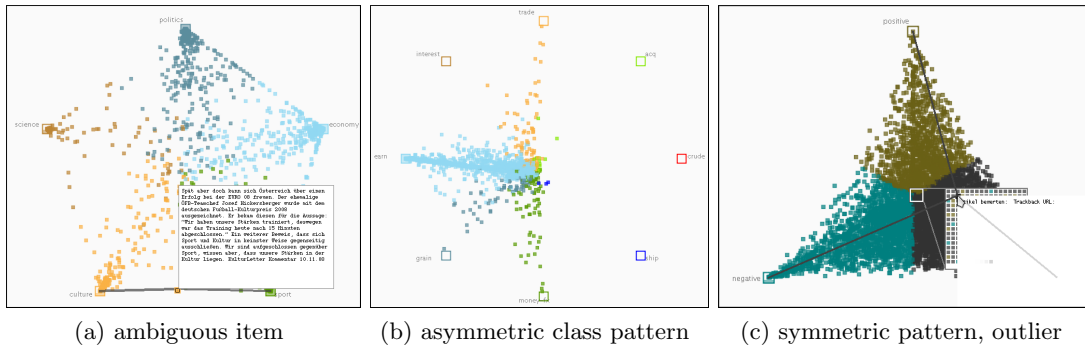


Figure 4.7: Examples visualizations revealing different phenomena: (a) competing classes and undecidable items, (b) dominant classes, and (c) outlier.

trainer getting a prize for his work in the cultural domain. From the text, the classifier can not decide whether the document belongs to the class “sports” or “culture” (which also seems no obvious decision for human labelers).

Further, one can select items for certain classes and investigate the item’s neighborhood. Suppose one would only work with a tabular user interface, and the items are sorted by their confidence. With this interface one can also easily access undecidable items but with less additional information. The part of information one gets depends on the sorting strategy of the table. Sorting might be implemented by class, by highest confidence value, or by entropy of the a-posteriori distribution.

4.1.8 Suitability for Visual Active Learning

This section investigates the suitability of the Class Radial Visualization for Visual Active Learning based on the requirements identified in section 3.4.4.

1. *Judge problematic behavior:* The Class Radial Visualization allows to judge problematic behavior, for instance a bias towards specific classes. (see class “earn” in figure 4.7b).
2. *Identification of problematic items:* The Class Radial Visualization allows to identify problematic items. (see figure 4.7c and 4.7a).
3. *Batch selection:* The Class Radial Visualization allows for batch selection in principle. For instance, lasso selection could be easily implemented. However, the current interaction mechanism only allow to select single items. For batch selection, the feedback mechanism to the classification model would need to be adapted, because lasso selection might select items that belong to different classes. For a detailed discussion of this, see the future work (section 6.2).

4 Implementation

4. *Classifier independence:* The Class Radial Visualization can be applied to any multi-class single-label classifier that outputs a-posteriori probabilities.

The Class Radial Visualization satisfies 3 of the 4 conditions for a visualization for Visual Active Learning. The condition *batch selection* is not satisfied in the current implementation. While batch selection could be easily be implemented in the visualization, the classifier backend would need major adaptations (see future work in section 6.2). However, this condition is more nice-to-have and would speed up the labeling process even more.

4.1.9 Comparison to Related Work

In this section the Model Uncertainty Visualization of the Visualix [LC09] and the Class Radial Visualization are compared to each other. Visualix has been identified as similar related work in section 3.1.3. An overview of the comparison of Visualix and the Class Radial Visualization is given in table 4.2.

Both are classifier-agnostic and represent the uncertainty of a classification model on the unlabeled data. However, the Visualix application uses confidence values as basis for the visualizations which are not necessarily a-posteriori probabilities (except for classifiers like the Naive Bayes). This means, the Model Uncertainty Visualizations of two different classifiers are, in general, not visually comparable. The items in the Visualix are colored according to a linear combination of class colors and according confidence values in the RGB color space. This schema of coloring allows to assess the uncertainty of the classifiers decision via the color of the item, but does not necessarily allow to assess the final decision of the classifier. On the contrary, the Class Radial Visualization uses the color of the most probable class as the color of the item. Here, the uncertainty of the decision is not reflected in the color of the item. This means, that in the Visualix visualization, the uncertainty of the decision is encoded twice – in the placement and the color of the item, but the final decision is not shown.

Summing up, compared to Visualix the Class Radial Visualization has the following advantages: (i) It allows to compare between visualizations of any two classifiers which is basis for a relative judgment. (ii) The classifiers final decision are encoded in the visualization.

4.1.10 Summary

In this section an interactive visualization, the Class Radial Visualization, was presented. This visualization is designed to visually assess and understand decisions and results of a classifier. It is based on a-posteriori probabilities and can therefore be applied to any classifier that outputs a-posteriori probabilities or whose output can be mapped to a-posteriori probabilities. A simple-to-use interaction mechanism allows users to adapt the classification model. Furthermore, extensions of the visualization were described in

Table 4.2: Comparison of the Model Uncertainty Visualization of the Visualix application [LC09] and the Class Radial Visualization

	Visualix	Class Radial Visualization
visualization space	3D	2D
data	confidence values	a-posteriori probabilities
type of problem	multi-class, single-label	multi-class, single-label
feedback	correct decisions by drag and drop	correct decisions by drag and drop
item coloring	linear combinations of class colors, weighted by confidence	color of most probable class
comparability	visualizations not comparable to each other	visualization comparable to each other

this chapter, the magic lens and the misclassification view. The magic lens view is a means to overcome the problem of overlapping items. The misclassification view allows for deeper understanding of the classification model – if labeled test data is available.

The suitability of the Class Radial Visualization for assessment and understanding of classifiers is shown in experiments in sections 5.2 and 5.5. An implementation of the concept of Visual Active Learning (see section 3.4) presented in section 5.3 uses the Class Radial Visualization to support users in selecting items for labeling.

4.2 Confusion Maps

Confusion maps are a simple tools to visualize the class confusion matrix or contingency table (see Sec. 2.4.2). In this matrix the rows correspond to the classifier's decisions (labels) and the columns correspond to the ground truth (target). Confusion maps for visualizing classifier properties were introduced in [LSGJ10]. Visualizing matrices as a heat map is nothing new, the original idea has been proposed by Bertin [Ber83] (a translation from the original French book from 1967) without using a computer display. Here we describe the application to a special matrix, the class confusion matrix. Further it is shown, why this visualization is suitable in the context of classification problems.

4.2.1 Method

The idea is derived from the heat map visualization [Ber83, WF09]. Where in the heat map visualization a value is represented by one pixel, in the Confusion Maps, each value is represented by a uniformly colored square. The color of a square is determined by linearly mapping the value range to a color palette, usually a palette ranging from white to any other color. This means the lowest (possible) value is mapped to the color white and the highest (possible) value to any other color. In RGB space the color (rgb) for a given value x is then be calculated as follows:

$$\begin{aligned} m_r &= \frac{r_{max} - r_{min}}{v_{max} - v_{min}} \\ m_g &= \frac{g_{max} - g_{min}}{v_{max} - v_{min}} \\ m_b &= \frac{b_{max} - b_{min}}{v_{max} - v_{min}} \\ r_x &= m_r \cdot x + r_{max} - m_r \cdot v_{max} \\ g_x &= m_g \cdot x + g_{max} - m_g \cdot v_{max} \\ b_x &= m_b \cdot x + b_{max} - m_b \cdot v_{max} \end{aligned}$$

where v_{max} is the maximum value to display, v_{min} the minimum value to display, $(r_{min}, b_{min}, g_{min})$ the RGB value of the color assigned to v_{min} and $(r_{max}, b_{max}, g_{max})$ the RGB value of the color assigned to v_{max} .

4.2.2 Parameters

Parameters of the visualization are the size of the squares (size of matrix cells) and the colors used for the highest and the lowest value, respectively.

4.2.3 Examples

Figure 4.8 shows the confusion matrix for a fully trained [Support Vector Machine \(SVM\)](#) classifier on the APA data set. As can be instantly seen the training set is perfectly classified (left). On the evaluation set (right) however, there are many off-diagonal (misclassified) elements. Especially, the first class has many items wrongly assigned. The huge difference of the Confusion Map on the training data set and the evaluation data set indicates that the classifier is not perfectly trained. Indeed, the pictures show results of a cross-domain experiment, where the classifier was trained on news articles and evaluated on blog entries which led to a decreased classifier performance.

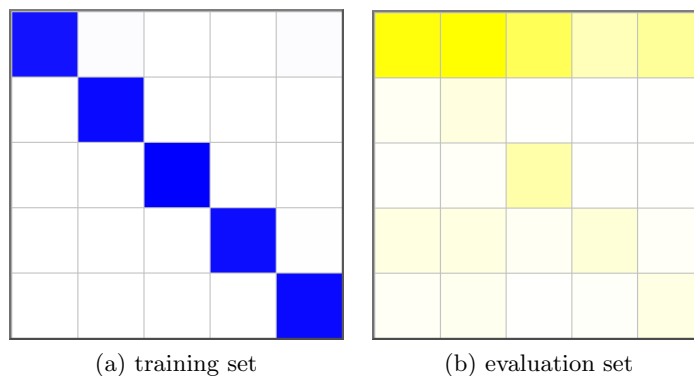


Figure 4.8: Confusion Maps. APA News Blog Data Set, 5 classes, trained with [SVM](#).

In figure 4.9 confusion matrices and Confusion Maps for a data set with more (19) classes are shown. In this example, the results on the training and the evaluation set are similar and only the visual representation can instantly reveal the locations of the differences (e.g., top right off-diagonal elements). However, the visualization does not allow to assess accurate values. For instance in the Confusion Map of the evaluation set (figure 4.8b), the two darkest values on the diagonal seem to be colored in the same gray, but as can be seen in the matrix have different values (48 and 51).

4.2.4 Properties

The Confusion Map visualization is suitable for getting an overview of the confusion matrix. It is especially helpful if the matrix is very large, such that single values can not be processed any more. Because the numbers are not shown and the human ability of distinguishing different hue values is limited [[CMS99](#)], the visualization can only show tendencies. However, certain patterns (e.g. a filled diagonal) and differences in the Confusion Maps for the training and evaluation data set can be easily detected.

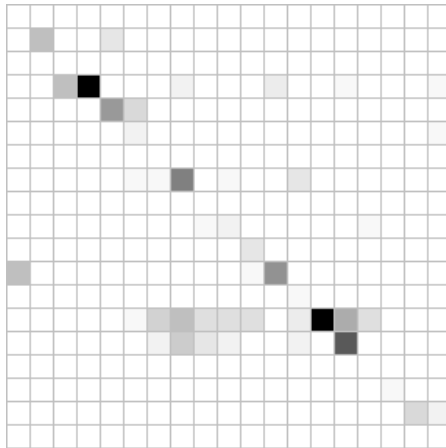
4 Implementation

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	10	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	10	40	0	0	0	2	0	0	0	3	0	0	1	0	0	1
0	0	0	0	16	6	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	20	0	1	0	0	4	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	1	17	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	1	7	10	5	5	5	0	3	40	13	5	0	0
0	0	0	0	0	0	2	8	4	2	0	0	2	0	26	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

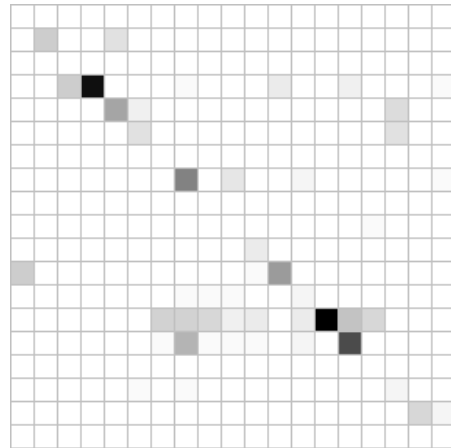
(a) confusion matrix, training set

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	10	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	10	48	0	0	0	1	0	0	0	4	0	0	3	0	0	1
0	0	0	0	18	3	0	0	0	0	0	0	0	0	0	7	0	0
0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	6	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	25	0	5	0	0	2	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	1	20	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	0	0	2	0	0	0	0	0
0	0	0	0	0	0	9	9	8	3	4	0	4	51	12	8	0	0
0	0	0	0	0	0	1	15	1	1	1	0	2	0	36	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(b) confusion matrix, evaluation set



(c) Confusion Map, training set



(d) Confusion Map, evaluation set

Figure 4.9: Comparison of confusion matrices and maps. Soybean data set (19 classes) [FA10], trained with KNN classifier (k=10)

4.2.5 Summary

This section presented the *Confusion Maps* visualization in detail. A Confusion Maps is a visual representation of class confusion matrix, allowing faster assessment and detection of patterns in the matrix. Confusion maps allow assessment of classifiers on the class level. Furthermore, aspects of classifiers, like over-fitting, can be visually identified by comparing Confusion Maps of training and evaluation sets. Confusion Maps have been implemented in the same application as the Class Radial Visualization (see section 4.1.3). Thus, the application covers all classifier-independent levels of classifier assessment (item, data set, class).

4.3 Tag Clouds

This section describes a family of tag layout¹ algorithms suitable for placing tags inside arbitrary, convex polygons without overlap. Placing tags inside a polygon can be reduced to placing the tags' bounding boxes, which is known to be a computational NP-hard problem (related to packing rectangles in irregular shapes). We apply a heuristics to find a good (not necessarily globally optimal) solution meeting the following criteria:

- lay out as many tags as possible,
- at least lay out the most important tags,
- avoid unnecessary white spaces between tags,
- avoid overlap between any two tags,
- use arbitrary, convex polygons as boundaries (i.e., not only rectangles, and including circles which can be approximated by polygons).

The special tag layout algorithm is used for the Voronoi Word Cloud visualization described in section 4.4 and applied in the experiments for classifier construction in section 5.4 as visual summary for texts. This tag layout algorithm has been published in [SKK⁺08] and includes an additional user study. Further, the tag layout visualization was implemented as module in the visualizing framework for the news article domain [LSKG08].

4.3.1 Method

In this section we describe the layout algorithm. First, we show possible ways to influence the tags' visual representation. Second we explain the calculation of the bounding boxes, and, third, we describe the core of the algorithm – the layout of the bounding boxes. Finally, we show possible combinations of the previous steps and obtain four different algorithms.

T is the set of all tags: $T = \{t_i | t_i \text{ is a tag, } 1 \leq i \leq n\}$. A tag t_i is given by a tag string t_i^s and a tag relevance value t_i^r (also called the weight of the tag). The algorithm assigns a certain position and font size to a subset of the tags, or ideally, all tags. We have no a-priory knowledge about the tags, however, there are some constraints on the layout: (i) the font size is not fixed, but has to be in a sensible interval (too small font sizes are not readable, too large font size range does not give a visually attractive layout) [HK07, Hof06], (ii) strings can be truncated, but have to consist of at least three letters, and (iii) the font family should be unique for all tags to not influence the perceived relevance of a tag. To account for these constraints we introduced three parameters for our algorithm: thresholds for the maximum and minimum font size allowed θ_{max} and θ_{min} , and an initial value for the font size assigned to the least relevant tag s_{min}^0 . The initial value for the font size for the most relevant tag, s_{max}^0 is set to θ_{max} . The current minimum and maximum font sizes s_{min}^0, s_{max}^0 define the current font range interval s_r^0 .

¹We use the term tag here from the application point of view, technically it can be an arbitrary string

There are two possible ways to change the size of the visual representation of a tag: (i) to shrink/enlarge the font size, i.e., the height and width of the tag, or (ii) to truncate the tag's string, i.e. to reduce the width of the tag. Given the input parameters θ_{max} , θ_{min} and s_{min}^0 we identified three processes to influence the tags' visual representation and therewith the bounding boxes of the tags: (i) shifting the font size interval, (ii) scaling the font size interval and (iii) truncating of tags' strings. If the initial layout trial was not successful (i.e. not all tags could be laid out with initial parameter settings), one of these processes changes the tags' visual representation leading to new, smaller bounding boxes. Another layout trial based on the new bounding boxes is then started. An overview of the algorithm is depicted in figure 4.10.

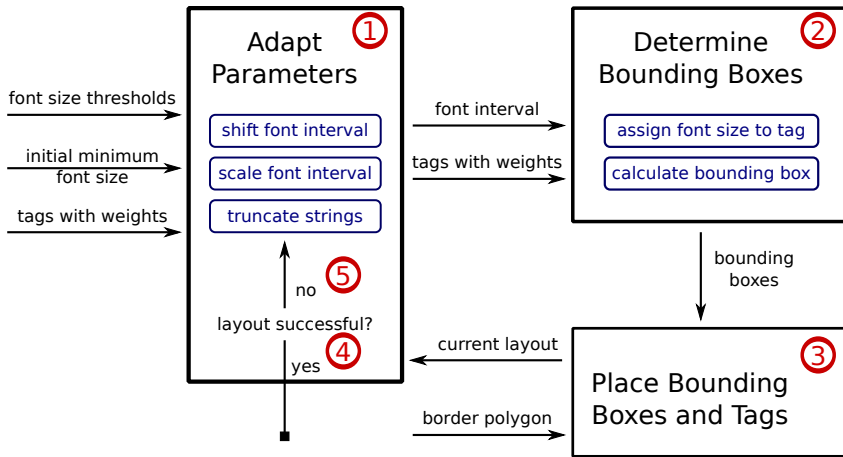


Figure 4.10: Algorithm Overview: ① Depending on input values and success rate of previous layout attempts a new font interval is determined by one of the three processes. ② The font size and the bounding boxes for all tags are calculated based on the current font interval. ③ Bounding boxes and tags are laid out inside the given border. ④ If all tags could be laid out or there are no more parameter changes possible the algorithm outputs the current layout. ⑤ If not, a new layout trial is started with the adapted parameter values.

We do not claim to achieve globally optimal layouts, we heuristically find a good layout that meets the following criteria:

1. The most relevant tags are laid out, i.e.: if a tag t_i is laid out, all tags t_j which could not be laid out have a lower or equal relevance value ($t_i^r \geq t_j^r$ for $t_i, t_j \in T$).
2. The tags are laid out comparable to an orbital layout, starting with the most relevant tag in the center of mass of the polygon. The center of mass was chosen, because it is defined for arbitrary polygons and yields consistent layouts over a wide range of shapes.
3. The bounding boxes of two distinct tags do not overlap.

4 Implementation

4. Two tags with the same relevance value are use the same font size.
5. A tag with a higher relevance value as another tag is drawn with an at least equal font size.

Influencing the Tags' Visual Representation

This section describes possibilities to adapt parameters after an unsuccessful layout trial. Section 4.3.1 then discusses possible combinations of these steps.

As mentioned above, the tags' visual representation can be changed by (i) shifting the font size interval, (ii) scaling the font size interval, and (iii) truncating the tags' strings.

The goal of the first method is the reduction of all font sizes in a linear way in order to get smaller bounding boxes. The interval shift is defined by: $s'_{min} = s_{min} - 1$ and $s'_{max} = s_{max} - 1$, which means the font size range keeps constant, $s'_r = s_r$. The minimum font size must not be less than the minimum font threshold: $s'_{min} \leq \theta_{min}$. Figure 4.11a depicts the procedure.

The scaling of the font size interval leads to a smaller range of font sizes by reducing the maximum value ($s'_{max} = s_{max} - 1$) while keeping the minimum value constant ($s'_{min} = s_{min}$). Therefore the font range decreases: $s'_r = s_r - 1$, with $s'_{max} \geq s'_{min}$ (see figure 4.11b).

The third method, the string truncation, substitutes the last letter of the tag string t_s with three dots. If this is done iteratively, the string becomes shorter while retaining as much information as possible (the beginning of the word and the indication, that the word continues). The minimum string length is set to 3. The truncation of strings leads to shorter strings and therefore to bounding boxes with smaller widths.

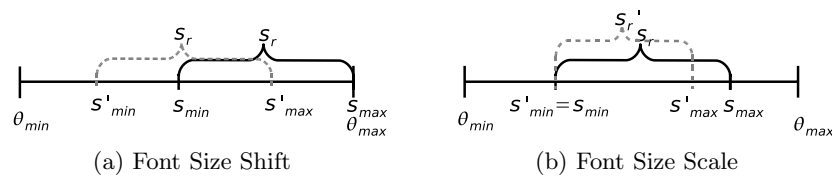


Figure 4.11: The font size interval can be shifted and scaled to decrease the size of the bounding boxes.

Determining the Tags' Bounding Boxes

This section describes how the bounding boxes are determined depending on the current font size settings and the tags' relevance value. First, each tag gets a font size value assigned and second, the bounding box for the current graphics display is determined from the tag's font size and the tag string. We are given the relevance values $t_i^r \in \mathbb{R}$

of the tags t_i and the maximum and minimum font size values s_{min} and s_{max} . We define a function m that maps tag relevance values to font sizes: $m : t_i^r \rightarrow t_i^s$, such that $s_{min} \leq t_i^s \leq s_{max}$, $1 \leq i \leq n$. Hoffmann [Hof06] identified three different font size distribution algorithms, the linear method, the logarithmic method and a clustering of font sizes. We implemented the linear and the logarithmic distribution. For the linear distribution, the font size t_k^s of the tag t_k is calculated as

$$t_k^s = (t_k^r - \min_i t_i^r) \cdot \frac{s_{max} - s_{min}}{\max_i t_i^r - \min_i t_i^r} + s_{min}$$

Accordingly, for the logarithmic distribution, the following equation is used

$$t_k^s = (\log t_k^r - \min_i \log t_i^r) \cdot \frac{s_{max} - s_{min}}{\max_i \log t_i^r - \min_i \log t_i^r} + s_{min}$$

Independently on which of the above methods was used, each tag $t_i \in T$ now has a valid font size t_i^s assigned. The width and the height of the bounding box t_i^{box} for the current graphics display is now determined by using library functions calculating a string's bounding box given a font size and a string.

Rectangle Placement

In this section we describe the core algorithm – the layout of rectangles in a convex polygon B . A rectangle $r = (w, h)$ is given by its width w and height h . R is the set of all rectangles $R = \{r_i | 1 \leq i \leq n\}$. The goal is to determine a subset $R' \subseteq R$, $|R'| = k \leq n$, and a set P of align points $P = \{p_i = (x_i, y_i) | 1 \leq i \leq k, p_i \text{ is the bottom left corner of } r_i\}$ such that

- all r_i are fully inside the polygon B ,
- no two r_i and r_j do overlap for $i \neq j$,
- R' contains all rectangles with the largest height, i.e. $h_i \geq h_j, \forall r_i = (w_i, h_i) \in R$ and $r_j = (w_j, h_j) \in R \setminus R'$.

The last condition arises from the overall goal to layout tags. In case, that not all rectangles would fit in the polygon, we want to drop the rectangles that represent the least relevant tags. The rectangles represent tags, their height is proportional to the tags' font size and therefore proportional to the relevance of the tag (see section 4.3.1). There are three more requirements to the layout algorithm arising from the tag layout application: (i) horizontal layout should be preferred to vertical layout to take the human westernized reading direction into account, (ii) compact layout is preferred, such that there is as little white space between boxes as possible, (iii) the most important tags should be in the center of the polygon as suggested by the notion of focus in [BZ05].

The algorithm is as follows: The rectangles are sorted descendingly by their height. The first rectangle is placed centered in the center of mass of the polygon. The edges of

4 Implementation

this rectangle split the polygon into four new polygons as depicted in figure 4.12a. The resulting polygons are stored in a list L together with the align point P and an align hint. The align point and the align hint are used for placing the next rectangle in the polygon. For the clipped *top*-polygon, the align hint will be *bottom* and the align point is the center point of the top-edge of the rectangle, see figure 4.12a for an example. Additionally, for each new polygon a priority value is stored. The priority v of a region is calculated from the distance d of its center of mass to the center of mass of the original polygon B and a weight value θ : $v = \frac{1}{d} \cdot \theta$, with

$$\theta = \begin{cases} 1, & \text{if region is top or bottom region} \\ 1.5, & \text{if region is left or right region} \end{cases}$$

This choice enforces the algorithm to create a compact and preferable line-by-line layout of the rectangles.

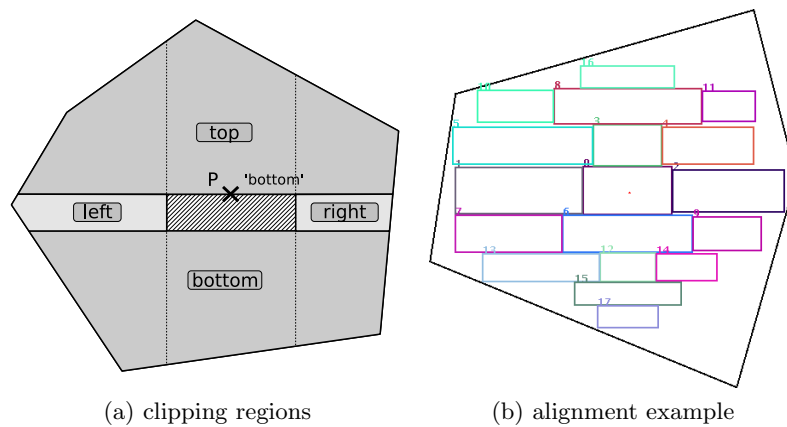


Figure 4.12: Rectangle Layout. (a) Clipping regions for the first rectangle, (b) 18 of 20 aligned random rectangles laid out in a polygon.

Four new polygons were created by placing the first rectangle. For each of the remaining rectangles (remember that the rectangles were sorted by height), all polygons with a larger area than this rectangle are processed in order of their priority value. When a rectangle could be placed in a polygon at the align point's location according to the align hint, zero to three new polygons are created and added to the list and the old polygon is removed from the list. In case the rectangle did not fit in the polygon the next polygon (next less priority value) gets tested. The algorithm stops, if for a rectangle all polygons from the list which at least the area as the rectangle were tested and the rectangle did not fit in one of these.

Combining the Steps

This section describes how the individual steps are combined into a system for tag layout. In figure 4.10 we gave a coarse overview of the whole procedure without describing the sequence of the individual steps. We identified four sensible possibilities to combine the steps (in a static way):

Shift-Trunc (ST): Try font interval shift first, if no success start to truncate tag strings. After one truncation reset the font interval. Try to lay out again. Go on with the interval shift. If no more shift possible, truncate again, and so on. Stop, if all tags are laid out or no more truncation is possible.

Shift-Trunc-Scale (STC): An extension of Shift-Trunc, which does not stop if no more truncation is possible, but tries to scale the font size interval instead. It stops if the interval can not be scaled anymore.

Shift-Scale-Trunc (SCT): Try font interval shift first, if no success try font interval scaling. If no success, try further with string truncation.

Trunc-Shift-Scale (TSC): Try string truncation first, if no success try font interval shifting. If still no success try font interval scaling.

As can be seen, the four proposed combination differ in the presence and sequence of their elements. As an example, Algorithm 2 shows the pseudo-code for the combination Trunc-Shift-Scale. The first operation of this algorithm (truncation) is inside the innermost loop. Thus, the algorithm tries to fit all tags by first truncating the tag strings as much as possible.

```

while not all boxes laid out and scaling possible do
  while not all boxes laid out and shifting possible do
    while not all boxes laid out and truncation possible do
      createBoundingBoxes()
      layOutBoxes()
      truncateTagStrings()
    end while
    shiftFontInterval()
  end while
  scaleFontInterval()
end while

```

Algorithm 2: Algorithm pseudo-code for Trunc-Shift-Scale (TSC)

4.3.2 Parameters





One parameters of the visualization is the boundary polygon (shape, size). Further parameters are the minimum initial font size, the maximum initial font size and the minimum allows font size. Good parameter settings for the font sizes can not be (easily) estimated automatically as they depend on the boundary polygon.

4.3.3 Examples

Figure 4.13 shows example tag clouds. In each column of the figure one layout algorithm has been applied to different tags and boundary polygons. In row (a) 10 tags were laid out in a polygon that approximates a circle. Row (b) shows the layouts for 138 tags from the del.icio.us website in a rectangular octagon. Row (c) shows the names of the European countries, the font size corresponding to the number of inhabitants. Row (d) shows 30 randomly chosen tags from the technorati website in a square. For all examples the parameters were set as follows: Minimum initial font size was set to 16 pt, maximum initial font size was set to 40, minimum allowed font size was set to 10. String truncation was only allowed until at least three characters of the string remained (excluding the dots). Table 4.3 shows the performance measures for all layouts of figure 4.13.

Algorithm (ST) fails to layout all tags in row (b) and (c). This is because the algorithm does not use scaling of the interval. However, the relative weight of the tags is better reflected by using this algorithm than by using any of the others. This fact is also reflected in table 4.3: using algorithm ST always results in different minimum and maximum font size. For the other algorithms, minimum and maximum font size may and usually does differ. Furthermore, it can be seen that the layouts differ little when the parameters and the bounding polygons are appropriately set (row (d) of the figure).

Table 4.3: Overview of the (technical) quality measures for the layouts of specific tag and border combinations as depicted in figure 4.13. (The table has the same arrangement in rows and columns as in the figure.)

Examples	ST	SCT	TSC	STC	
	100	100	100	100	tags laid out in %
	78	36	61	78	area filled in %
	22	0	39	22	truncated characters
	17...34	17...34	17...34	17...34	font interval in pt
	42	64	76	67	tags laid out in %
	76	79	78	78	area filled in %
	2	6	266	86	truncated characters
	10...34	10...10	10...10	10...10	font interval in pt
	11	100	100	100	tags laid out in %
	31	43	47	47	area filled in %
	10	0	73	75	truncated characters
	10...34	10...13	10...20	10...19	font interval in pt
	100	100	100	100	tags laid out in %
	68	63	74	68	area filled in %
	10	0	76	10	truncated characters
	10...34	10...34	12...36	10...34	font interval in pt

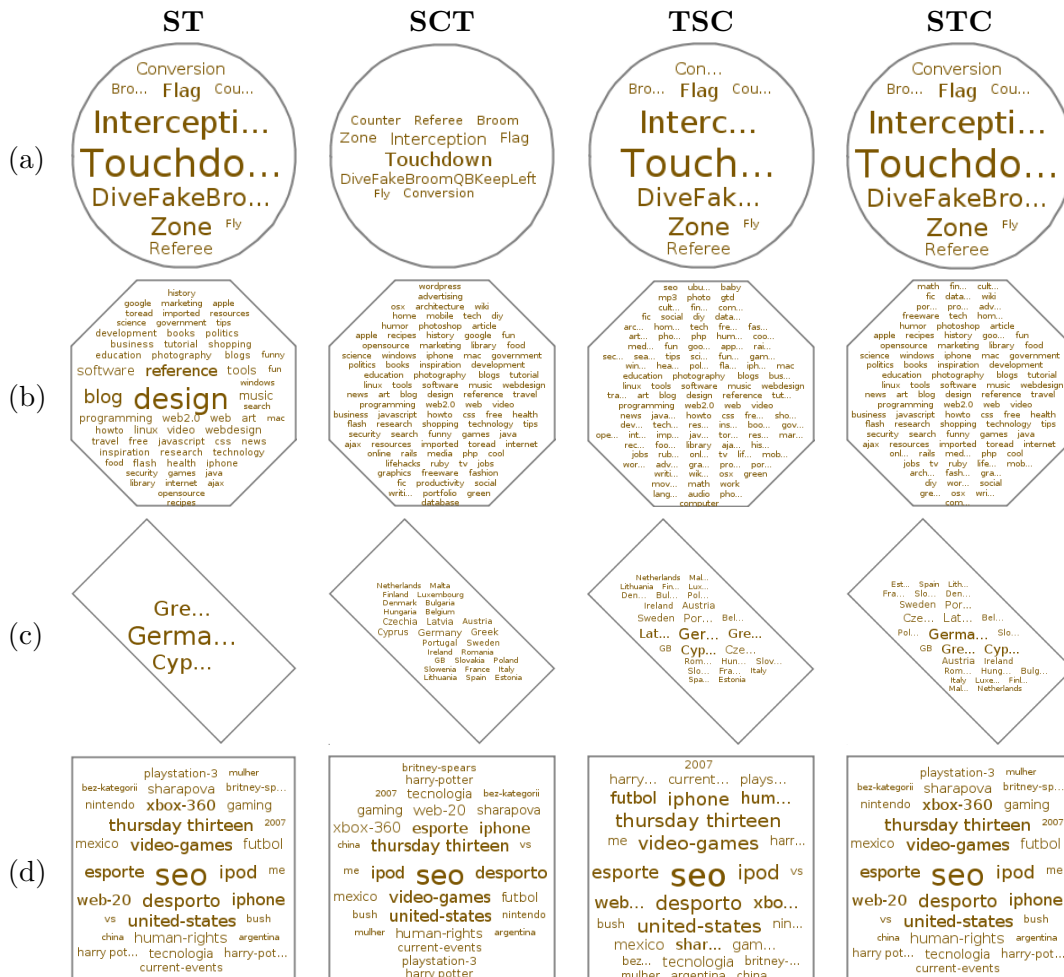


Figure 4.13: Examples of tag layouts produced by the four algorithms Shift-Trunc (ST, first column), Shift-Scale-Trunc (SCT, second column), Trunc-Shift-Scale (TSC, third column), Shift-Trunc-Scale (fourth column) with different tags and different borders: (a) 10 tags in a polygon approximating a circle, (b) 138 tags from del.icio.us in a regular octagon, (c) EU states weighted by the number of inhabitants in a rectangle, (d) 30 tags from technorati in a square.

4.3.4 Properties

The tag layout has the following properties:

- Tags can be laid out in arbitrary, convex polygons.
- The most important tag is placed in the center-of-mass of the polygon. All other tags are laid out in a circular manner around the most important tag in descending importance.
- It can not be guaranteed that all tags are laid out, but it can be guaranteed that the ones that are laid out are the most important ones.
- There is no overlap between any two tags.
- No unnecessary white spaces occur between any two tags.
- The runtime of the tag layout algorithm heavily depends on the initial settings of the parameters.

4.3.5 Summary

This section presented a family of tag layout algorithm suitable for producing compact tag layouts in arbitrary, convex polygons. The layout algorithms can be applied to any set of weighted words. The layout is performed using heuristics and not necessarily globally optimal. The algorithm tries to fit the next most important tag in the available area next to already laid out tags. If the tag fits, the area is cropped along the tag's bounding box. The next tag is then tried to be placed in the remaining area. In this iterative process it might occur that different cropped nearby areas may yield place for another tag but are not considered because they are threatened separately by the algorithm. Thus, further improvements to the algorithm include joining together these areas.

The special tag layout algorithm is used for the Voronoi Word Cloud visualization described in section 4.4 and used in the experiments for classifier construction in section 5.4 on page 131 as representation of text summaries.

4.4 Voronoi Word Clouds

The Voronoi Word Cloud visualization is an extension of the tag cloud visualization described in Section 4.3. It has been published in [SKG11]. The idea of this visualization is to represent not only one tag cloud, but multiple tag clouds and their relationships. Or in other words, the visualization can be used to represent multiple sets of documents, showing the tags or most important words for each set of documents.

Each set of documents D_i is characterized by a set of words W_i . These words can for example be tags associated to the set, key words automatically extracted from the documents or extracted named entities. Further, a similarity relationship between document sets can be displayed in this visualization. Therefore, for each pair of document sets (D_i, D_j) a similarity value needs to be given. This similarity value can for example be the cosine similarity of the median document vectors for both sets or a user given value for the semantic similarity of the sets.

4.4.1 Method

This section describes the method in general. An example use case for visualizing a text classification model is described in section 5.6.

1. For each pair of documents (D_i, D_k) a similarity value is calculated (for instance as the cosine similarity of the average document vector in the vector-space representation).
2. Each document set (D_i) is mapped into the 2D-space using a FDP algorithm. The attractive forces between two document sets are set to their similarity value. As a result for each D_i a corresponding point p_i in the 2D-plane is generated.
3. A Voronoi diagram is calculated using the points p_i as generator points obtaining a polygon P_i for each document set.
4. For each document set D_i the associated set of words W_i is calculated. This can for instance be all named entities mentioned in all documents. Optionally, a weight may be added to each word of W_i .
5. The weighted words for document set D_i are laid out inside the polygon P_i using the tag layout algorithm described in section 4.3.
6. For a similarity-based coloring of the polygons P_i each document set is projected into the 3D-space (similarly to step 2.), and their projection is used to select the background color for polygon P_i from the RGB color space according to formulas 4.1, 4.2 and 4.3.

$$r = \min(255, 255 \cdot (s + sx)) \quad (4.1)$$

$$g = \min(255, 255 \cdot (s + sy)) \quad (4.2)$$

$$b = \min(255, 255 \cdot (s + sz)) \quad (4.3)$$

4 Implementation

The scaling parameter s in the equations 4.1 to 4.3 is used to achieve a lighter background color to keep the foreground text readable. In the experiments s was set to 0.7.

4.4.2 Parameters

Parameters of this visualization are the number of words to display for each document set, the specific tag layout algorithm and its parameters. In an interactive visualization the number of words per document set could also be automatically adapted based on the zoom level. Further, the similarity measure for comparing document sets can be chosen. Also the formula for coloring the polygons can be substituted and might be due to future research to get colors that better reflect the similarity for the human eye. The mathematical distance in the RGB color space is not proportional to perceived similarity of colors.

4.4.3 Examples

Figure 4.14 shows an example Voronoi Word Cloud representing three document sets. The borders are set to a regular dodecagon. The feature vectors for the sets were set as $f_1 = (1, 0, 0, 0)^T$, $f_2 = (0, 1, 0, 0)^T$, $f_3 = (1, 1, 1, 0)^T$. This means, document set 1 and 2 show no similarities and document set 3 is equally similar to the other twos.

The words were weighted by $w_{ij} = \frac{i}{j}$, for $i = 1, 2, 3$ indicating the document set and $1 \leq j \leq 10$ indicating the word index. The string of the word was simply set to "word" for $i \bmod 3 \neq 2$, and "long_word" for each third word. The region for document set 1 is colored yellow, for document set 2 is colored blue, colors that are at far apart in the RGB color space. The region of document set 3 is colored green, a color that is a mixture of blue and yellow. This coloring reflects the similarity of the document sets.

4.4.4 Properties

By the process of construction, the Voronoi Word Cloud visualization has the following properties. First, similar document sets are placed near each other. Second, similar document sets are colored in similar colors (in RGB) space. Third, a document set is represented by a set of multiple words. Note that the importance of the words can not be directly compared across different document sets. The same absolute size does not mean the same global importance. This is because the tag layout is done independently in each polygon. However, the relative size of words can be compared across document sets, i.e. the larger the word, the more important it is for the specific set.

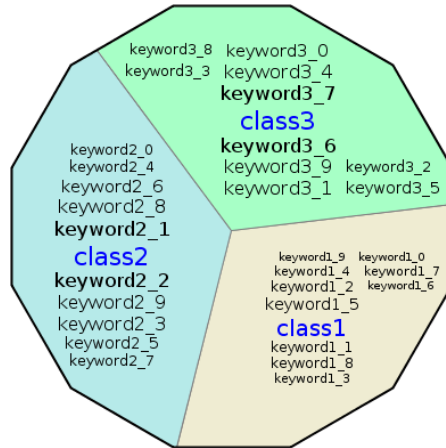


Figure 4.14: Example Voronoi Word Cloud for 3 document sets and 10 words per document set.

4.4.5 Summary

In this section a visualization called, Voronoi Word Clouds, for multiple sets of documents is described. This visualization combines Voronoi subdivision and the tag layout for arbitrary convex polygons described in section 4.3. Voronoi Word Clouds can be used to reflect the relatedness between documents sets (by the spatial layout) and an overview of the content for each set. The experiment in section 5.6 applies the Voronoi Word Clouds to visualize the model of a specific text classifier.

4.5 Summary

In this chapter four different visualizations were described. The experiments presented in the next chapter answer the research question of this thesis using these visualizations and the concept of Visual Active Learning.

In section 4.1 an interactive visualization, the *Class Radial Visualization*, is described. This visualization conveys classification results based on a-posteriori probabilities. This visualization is classifier-agnostic and can be applied to any classifier that outputs a-posteriori probabilities or whose output can be mapped to a-posteriori probabilities. Further, an integration in an application was described. The Class Radial Visualization was designed to support assessment, understanding and adaptation of classifiers. The experiments described in section 5.2 evaluate the aspect of assessment and understanding, the experiments in section 5.3 and 5.5 evaluate the aspect of adaptation.

In section 4.2 the *Confusion Maps* visualization was presented. Confusion maps are a visual representation of class confusion matrices, allowing faster assessment and detection of patterns in the class confusion matrix. Further, aspects of classifiers, like over-fitting, can be visually identified by comparing Confusion Maps of training and evaluation sets.

Section 4.3 describes a special layout algorithm for *tag clouds* in arbitrary, convex polygons. This visualization is the basis for the experiments in section 5.4 dealing with fast classifier construction for text classification.

The tag layout algorithm is applied in the *Voronoi Word Cloud* visualization presented in section 4.4. This visualization is applied to a special text classifier in the experiments in section 5.6 in order to allow detailed assessment of this specific classifier.

'It is a capital mistake to theorize before you have all the evidence. It biases the judgment.'

(Sherlock Holmes in A Study in Scarlet)

5 Experiments

This chapter describes the experiments performed to answer the general research question of the thesis (see section 1.1):

RQ 'Can interactive visualization improve construction, understanding, assessment, and adaptation of supervised machine learning algorithms?'

First, the proposed visualizations Class Radial Visualization and Confusion Maps (see section 4.1 and 4.2) are applied on different data sets by expert users in an exploratory way (**Experiment Set 1**). Results show, that visualizations are suitable for experts to assess and understand arbitrary classifiers.

Having covered the assessment and understanding part of the research question in Experiment Set 1, **Experiment Set 2** tackles the adaptation part. In these experiments, in (i) a small user study, and in (ii) more extensive user simulations Visual Active Learning is compared to active learning. More specifically, it is evaluated whether the selection of training data by the user outperforms random and automatic approaches. These experiments do not take the time in to account that is required for labeling the data items.

However, specifically for text classification the labeling time has the most influence on the total time of training data generation. Thus, in **Experiment 3** alternative representations of text documents are compared in a user study for their suitability to efficient labeling. The findings relate to the adaptation and generation part of the research question.

Until now, in all experiments it was assumed that the classes are known beforehand. In most applications this is a feasible assumption, however there are some cases where the classes only emerge while the users explore a data set. The latter case is explored in the **Experiment 4**. This experiment is to be considered a proof-of-concept. Starting with a pure data visualization the user can interact with the data and explore potential classes and construct the classifier from scratch.

Having considered mostly the domain of text classification in previous experiments, **Experiment 5** further investigates the understanding of a specific text classifier.

5 Experiments

More specifically with the experiments presented in this chapter the following questions are answered:

1. Can visualizations, more specifically the Class Radial Visualization and the Confusion Maps help experts to assess and understand arbitrary classification models? **Experiment Set 1** → see Sec 5.2.
2. Can user feedback on classification models through interactive visualizations, more specifically an interactive version of the Class Radial Visualization, be used to improve classification models? Is there a benefit over automatic methods? **Experiment Set 2** → see Sec 5.3
3. What are good representations of the data to classify, more specifically of text documents, to speed-up the manual labeling process? **Experiment 3** → see Sec 5.4
4. Can pure data visualizations be used to allow domain experts to generate their own classifiers? **Experiment 4** → see Sec 5.5
5. In which way can a model of a specific text classifier, namely the **Class-Feature-Centroid (CFC)** classifier, be visualized and made accessible to users? **Experiment 5** → see Sec 5.6

Section 5.1 gives an overview of the data sets used in the experiments. Table 5.1 summarizes the experiments in this chapter by their (i) topics (the aspects of the research question), (ii) the visualization modules used, (iii) the methodology used in the experiment, and (iv) the results.

Table 5.1: Overview of all experiments in this chapter. Column 2 shows the aspect of the research question the experiment covers, column 3 names the visualization modules as introduced in chapter 4, column 4 shows the experimental procedure and column 5 summarizes the results.

Experiment	Topics	Modules	Methodology	Results
Experiment Set 1 Classification of images and text →Sec 5.2	understanding, adaptation	Class Radial Visualization, Confusion Maps	explorative, expert users	understandin, unexpected insights with visualization
Experiment 2 Visual Active Learning →Sec 5.3	generation, adaptation	Class Radial Visualization	user simulations	Visual Active Learning can outperform classical active learning
Experiment 3 Document representation for efficient labeling →Sec 5.4	generation	Word Cloud	user study	word clouds are efficient representation for topical labeling
Experiment 4 Visual hypothesis generation →Sec 5.5	generation, adaptation	Class Radial Visualization, Information Landscape	prototype	proof-of-concept, classifier generation from scratch
Experiment 5 Understanding model of a text classifier →Sec 5.6	understanding	Voronoi Word Cloud	prototype	proof-of-concept

5.1 Data Sets

In this section the data sets use in the experiments are presented in detail. Section 5.1.1 discusses the famous Iris flower data set, section 5.1.2 the COIL-20 data set, a well known data set for image classification. Three text data sets are presented in sections 5.1.3 (Reuters-21578), 5.1.4 (APA Blog and News) and 5.1.5 (20 Newsgroup).

5.1.1 Iris Flower Data Set

The Iris flower data set is very famous and frequently referenced in the pattern recognition literature (see for instance [WF05]). The data set was initially published by R. Fischer in 1936 [Fis36]. The classification task is to distinguish three different types of the Iris flower: Iris Setosa, Iris Versicolor and Iris Virginica. Each flower is represented by four attributes, namely (i) sepal length in cm, (ii) sepal width in cm, (iii) petal length in cm, and (iv) petal width in cm. The class Iris Setosa is linearly separable from the other two classes, whereas the classes Iris Versicolor and Iris Virginica are not linearly separable from each other. For the experiments the data set was randomly split into train (34%), test (33%) and evaluation set (33%) as shown in table 5.2.

Table 5.2: Overview of the 34:33:33 split of the Iris Flower data set. Showing the number of instances in each set.

class (flower)	total	train	test	eval
Iris Setosa	58	20	18	20
Iris Versicolor	50	19	18	13
Iris Virginica	42	12	14	16
total	150	51	50	49

5.1.2 COIL-20 Image Data Set

The Columbia Object Image Library (COIL-20) [NNM96] consists of images of 20 different objects. For each object 72 different images were taken using a turntable yielding 1440 images in total. Each image of an object shows the object from another angle, while the angle is increased by 5 degrees in subsequent views. Figure 5.1 shows the 20 objects from a 0 degree angle. In the experiments the processed version of the images are used, where the background has been discarded and the images are cropped to the smallest square which contains the object. Different splits of the data set were generated as shown in table 5.3. For the 50:50 split the even view numbers serve as train images, i.e. all 0,10,20,.. degree images are contained in the training data set.



Figure 5.1: Objects of the COIL-20 image data set

Table 5.3: Overview of the splits of the COIL-20 data set

split	train	test	eval	total
50/50 odd-even	720	720	0	1440
60/20/20 random	864	289	287	1440
34/33/33 random	490	476	474	1440

5.1.3 Reuters-21578 Text Corpus

The Reuters-21578 data set¹ is a standard data set for text classification tasks. The data set contains news articles published by Reuters in 1987. The news articles were manually categorized by publisher. The full data set is a multi-label data set. This work focuses on single-label classification tasks and therefore the single-label R8 category subset was used. This subset contains only documents with a single topic and classes which still have at least 2 documents (one for training, one for test). The data set was split into train (60%), test (20%), and evaluation set (20%). Table 5.4 gives an overview of the R8 subset and the splits used in the experiments. The data was vectorized as described in section 2.3, the dictionary for the noun-vector-space contains 23,534 nouns in total.

¹<http://www.daviddlewis.com/resources/testcollections/reuters21578/>

Table 5.4: Overview of the R8 subset of the Reuters-21578 data set

class	total	train	test	eval
acq	2394	1454	487	453
crude	510	320	93	97
earn	3953	2328	795	830
grain	566	349	119	98
interest	306	158	77	71
money-fx	553	334	100	119
ship	182	114	35	33
trade	473	305	81	87
total	8937	5362	1787	1788

5.1.4 APA Blog and News Corpus

This data set is designed for cross-domain classification task and consists of news articles collected by the Austrian Press Agency (APA) and related blog posts crawled from the World Wide Web. The details of this data set were published in [LSGJ09b, LSGJ09a, LSGJ10].

The news corpus contains 27,570 documents, the blog corpus contains 10,977 documents. Each news article is labeled with one of the five categories 'sports', 'culture', 'politics', 'economy', 'science'. Each class consists of ≈ 5500 documents, thus the corpus is nearly balanced (see table 5.5).

The blog corpus contains posts of 56 blogs which were selected according to the given newspaper categories: 10 politics blogs, 10 economy blogs, 10 sports blogs, 11 culture blogs, and 15 science blogs. Each blog post is labeled by one of the news categories, according to the labeling of the whole blog. The number of blog posts for each category can be seen in table 5.6. Because the blog posts were labeled with the category of the blog, all entries of a single blog belong to the same category. The blogs were randomly checked whether this assignment is true. Not all blogs and blog posts were examined manually, thus it can not be ensured that there is no mislabeled blog post. Because of the potential mislabelings in the training data the resulting accuracy of a classifier is limited by a number less than 100%.

The news articles and blog entries were preprocessed as outlined in section 2.3 on page 25. Two noun-vector spaces was created using (i) [Term Frequency - Inverse Document Frequency \(TF-IDF\)](#) weighting and (ii) [Okapi BM25 ranking \(BM25\)](#). The news data set contains $\approx 237,000$ nouns with an average document length of 92.5 nouns. The blog data set contains $\approx 110,000$ nouns with an average document length of 61.5 nouns. The

merged dictionary consists of $\approx 302,000$ different terms, the sum of the distinct terms in the news and blogs dictionary is $\approx 347,000$. This means, that the news and the blog corpus share only $\approx 45,000$ terms.

Both corpora were randomly split into training, test and evaluation set, whereas the training set contains $\approx 60\%$, the test set $\approx 20\%$, and the evaluation set $\approx 20\%$ of the documents. The detailed numbers can be found in table 5.5 for the news corpus and in table 5.6 for the blog corpus.

Table 5.5: Overview of the APA News Topic Dataset

class (topic)	all	train	test	eval
science	5556	3349	1158	1049
politics	5302	3173	1077	1052
sports	5595	3421	1058	1116
culture	5548	3267	1132	1149
economy	5569	3333	1087	1149
total	27570	16543	5512	5515

Table 5.6: Overview of the APA Blog Topic Dataset

class (topic)	all	train	test	eval
science	1411	871	273	267
politics	2786	1657	565	564
sports	2478	1480	495	503
culture	1106	638	241	227
economy	3196	1941	620	635
total	10,977	6587	2194	2196

5.1.5 20 Newsgroup Corpus

The 20 Newsgroup data set² is well known in the text classification and clustering literature. It consists of newsgroup posts (netnews) organized in 20 different newsgroups (classes). The posts are nearly uniformly distributed across the classes (see table 5.7). Some of the newsgroups are topically related to each other, for instance

²<http://people.csail.mit.edu/jrennie/20Newsgroups>

5 Experiments

talk.religion.misc and soc.religion.christian, while others are topically unrelated, for instance talk.politics.guns and sci.crypt. In the experiments, the original data set is used which contains 19.997 newsgroup posts in total.

Table 5.7: Overview of the 20 Newsgroup data set

class (newsgroup)	all	train	test	eval
alt.atheism	1000	349	339	312
comp.graphics	1000	320	346	334
comp.os.ms-windows.misc	1000	323	321	356
comp.sys.ibm.pc.hardware	1000	341	319	340
comp.sys.mac.hardware	1000	349	335	316
comp.windows.x	1000	331	331	338
misc.forsale	1000	348	328	324
rec.autos	1000	346	317	337
rec.motorcycles	1000	333	343	324
rec.sport.baseball	1000	322	330	348
rec.sport.hockey	1000	368	305	327
sci.crypt	1000	327	333	340
sci.electronics	1000	344	318	338
sci.med	1000	331	319	350
sci.space	1000	359	311	330
soc.religion.christian	997	321	345	331
talk.politics.guns	1000	340	327	333
talk.politics.misc	1000	362	342	296
talk.politics.mideast	1000	349	333	318
talk.religion.misc	1000	336	357	307
total	19,997	6799	6599	6599

5.2 Experiment Set 1: Quality Assessment of Classifiers

The first set of experiments considers the understanding part of the research question. These experiments investigate whether two of the developed visualizations, namely the Class Radial Visualization and the Confusion Maps, may aid expert users in understanding classification models.

More specifically the experiments try to answer the following question:

Can the Class Radial Visualization and the Confusion Maps help experts to assess and understand arbitrary classification models?

The experiments presented in this section differ in the analyzed data set. In the first experiment (see section 5.2.1), an image data set was used. The second experiment in section 5.2.2 investigates a text classification scenario. In both cases the users applying the visualization were machine learning experts – namely all co-authors of the papers where the experiments have been published. A shorter version of the experiments for the image classification task described in section 5.2.1 was published in [SL09a]. The experiments on the cross-domain text corpus described in section 5.2.2 were published in in [LSGJ09b, LSGJ09a, LSGJ10].

5.2.1 Quality Assessment for Image Classification

This experiment investigates whether the Class Radial Visualization may aid machine learning experts in understanding classifier behavior on an image data set. Classically, experts would refer to various performance measures of trained classifiers. Here we will investigate if experts could benefit from additional visualizations.

Procedure

For the first experiment an object recognition task was chosen. A classifier should predict object labels for unseen images given a set of training images that contain different objects. The COIL-20 image database was used for this purpose (see section 5.1.2 for a description of the data set). Two different feature transformation methods were tested: [Principal Component Analysis \(PCA\)](#) [TP91] and [Fisher-Linear Discriminant Analysis \(LDA\)](#) [BHK97] and two different classification models: [k-Nearest Neighbor \(KNN\)](#) with $k = 10$ and [Naive Bayes \(NB\)](#).

For the feature transformation methods, [LDA](#) features are known to outperform [PCA](#) features for object recognition tasks. Regarding the classifiers, [KNN](#) and [NB](#) perform similar in general, it depends on the data at hand which model yields better results [Kot07]. The aim of the experiment is to find out whether those facts are reflected in the visualization. We assume that with our visualization we gain insights into the

5 Experiments

classifier performance as well as the classification process. For the classifiers, we used the implementation of the WEKA machine learning library [WF05].

For each of the four scenarios – (i) **PCA** features and **KNN** classifier, (ii) **PCA** features and **NB** classifier, (iii) **LDA** features and **KNN** classifier, and (iv) **LDA** features and **NB** classifier – we set the step size to 72 items (10% of the training set) and apply our procedure stepwise until the classifier has seen all trainings items.

Results

The screenshots in figures 5.2 and 5.3 show the resulting visualization for all combinations for the fully trained classifier.

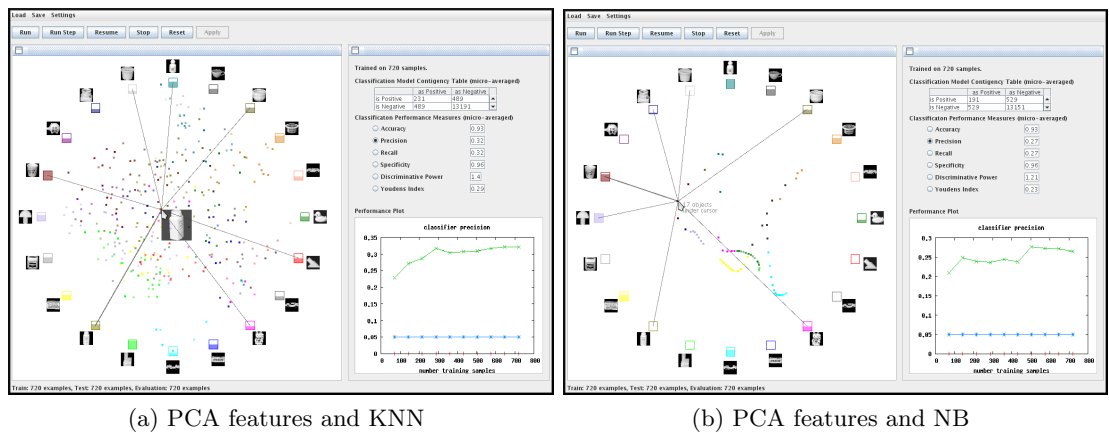


Figure 5.2: Classifier visualization on PCA features for the COIL-20 database

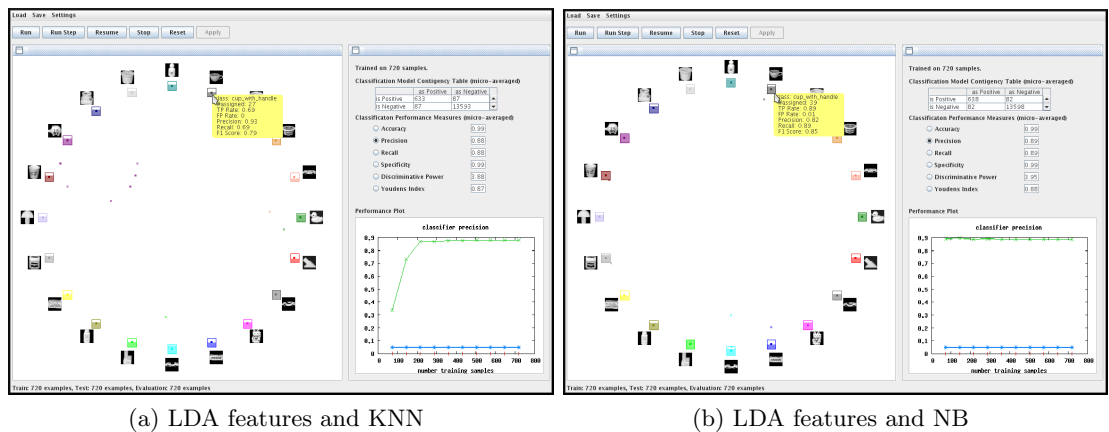


Figure 5.3: Classifier visualization on LDA features for the COIL-20 data set

5.2 Experiment Set 1: Quality Assessment of Classifiers

It is obvious that the **KNN** classifier cannot learn a reliable model using the **PCA** features (figure 5.2a). The items are widely distributed and the colors are mixed. This indicates that the classifier cannot confidently predict most of the items. In figure 5.2a an item is selected and via mouse over the original data and the class probabilities are shown. A user can instantly see that the classifier was not able to correctly assign this item to one class because seven lines are drawn. The visual information on classifier level correlates well with the measures calculated on the evaluation set (given on the right-hand side of the figure). A precision of only 0.32 is reached for this trial.

Figure 5.2b shows the result for the **NB** classifier on the **PCA** feature set. Although there seems to be less chaos in the Class Radial Visualization compared to the **KNN** case, the **NB** classifier also has a low precision value of 0.27. In the **NB** case, the visual information on classifier level does not correlate well with the performance measure results. Therefore, to correctly assess the classifier performance it is necessary to also consider the class level and the test item level: It can be seen that the squares of nine out of 20 classes are not filled at all. This means that no test item is labeled with these classes. Also, many test items are mapped to the same location which can be seen via mouse over.

In contrary, the classification based on **LDA** features generates a completely different layout (see figure 5.3). Almost all test items are placed near their corresponding class items and the center of the circle is empty. Both **KNN** and **NB** perform well on the **LDA** features. This can be observed in the Class Radial Visualization as well as in the performance table. The precision value is 0.88 for **KNN** and 0.89 for **NB**. From the performance plots on the right-hand sides of figures 5.3a and 5.3b we conclude that **NB** needs less training items (72 items) than **KNN** (210 items) to reach the final precision on the evaluation set.

Figures 5.3a and 5.3b also show the detailed information for a single class. Obviously the **KNN** missed some of the objects 'cup-with-handle" (**True Positives (TP)**-rate = 0.69) but made no false assignment (**False Positives (FP)**-rate=0). The **NB** classifier found more correct objects 'cup with handle" (**TP**-rate = 0.89) but classified at least one item incorrectly as 'cup-with-handle" (**FP**-rate=0.01).

Discussion

To sum up, the experiments revealed that the visualization clearly reflects the difference between **PCA** and **LDA** features for the used data set. The visualization also clarified that the **KNN** and the **NB** classifier perform quite equally on the **LDA** feature set. Furthermore, from the visualization we were able to conclude that the **NB** needs less training items than **KNN**. However, we also found that an interactive investigation in the visualization is necessary to assess the classifier correctly.

5 Experiments

Summary

This experiment investigated whether the Class Radial Visualization can aid machine learning experts in understanding classifier behavior. We applied the Class Radial Visualization on an image data set on two different feature-spaces with two different classifiers and showed which conclusion could be drawn from the visualization – beyond the conclusions obvious from evaluation measures.

In one sentence the outcome of this experiment can be summarized as follows:

The Class Radial Visualization can help machine learning experts to assess and understand arbitrary classification models for image classification tasks.

5.2.2 Quality Assessment for Cross-Domain Text Classification

This experiment investigates whether the Class Radial Visualization may aid machine learning experts in understanding classifier behavior on a text data set. The chosen classification task is a cross-domain task. In cases without training data available for a domain a possible approach is to train on a domain which exhibits similar features and characteristics. However, it is not clear how a trained model generated on one domain would perform on another domain. Correctly applied machine learning algorithms avoid over-fitting on the training set to maximize the generalization capabilities. In case of cross-domain text classification, an implicit fitting to the training corpus vocabulary is unavoidable.

The cross-domain classification task is to classify blogs into commonly agreed upon newspaper categories, where only training data from the news article domain is available.

Given the labeled news articles and the unlabeled blog corpus, the question is: Can this data be used to apply high quality cross-domain classification from news to blogs? Further, the question arises whether the classification can be performed in a fast and efficient way, because the corpora are highly dynamic and grow daily.

Several text classification algorithms are applied on the problem setting and the performance of these algorithms is evaluated for different scenarios. We claim that the generalization abilities of text classification algorithms are sufficient when the classifiers implicitly concentrate only on the most important text features weighted with state-of-the-art techniques. For a visual evaluation, we use the classification visualization described in section 4.1

Procedure

The classification task is a cross-domain multi-class problem with five classes. The corpus consists of two sub corpora. The first corpus consists of news articles, which have been manually labeled with one out of five news category. The second corpus, further referred to as blog corpus, was crawled from the World Wide Web. For a detailed description of the data sets see section 5.1.4.

Because the classification task is highly dynamic (daily changing news articles and blogs), a classifier with low computational complexity is desirable. Centroid based classifiers are known to achieve good results in terms of accuracy and time complexity [HK00, LT04]. We chose a relatively new centroid-based text classifier, the CFC (see section 2.2.6) which has been reported to be extremely fast and outperform Support Vector Machines (SVMs) and all other centroid based text classifiers. For the cross-domain classification task, the CFC classifier, and two standard text classifiers were compared: the KNN (see section 2.2.3) and SVM (see section 2.2.5), namely the LibLinear implementation with a linear kernel [FCH⁺08]. As outlined by Sebastiani [Seb02], SVM and KNN classifiers are the best performing standard text classification algorithms.

Experimental Settings For a description of the APA news and blog data set see section 5.1.4. For the visualization and the evaluation of our classification task, we split the data sets into a fixed training and test set. We randomly selected 70% of the data as training set and 30% as test set. To measure the performance of the classifiers the algorithms were evaluated in the following four scenarios:

NewsNews: The training set of the news corpus has been used to train the classifiers and we report the performance on the news evaluation set.

BlogBlog: The training set of the blog corpus has been used to train the classifiers and we report the performance on the blog evaluation set.

NewsBlog: The training set of the news corpus has been used to train the classifiers and we report the performance on the blog evaluation set.

BlogNews: The training set of the blog corpus has been used to train the classifiers and we report the performance on the news evaluation set.

For a weighting schema, we used **BM25** [JWR00] for **KNN** and **SVM** with the standard parameters $k = 2$ and $b = 0.75$. We also experimented with variants of **TF-IDF** for both algorithms, yet the algorithms performed best with **BM25**. For **CFC**, we used a standard **TF-IDF** weighting, as recommended by the authors. We also tested the algorithm with **BM25**, but the results got worse than with **TF-IDF** weighting, as expected.

For the **KNN** algorithm, we conducted a manual parameter search and identified $k = 10$ to be the best parameter setting. For the **SVM**, we used a linear kernel which is reported to outperform non-linear kernels in text classification [YL99]. We also experimented with various values for parameter b in the **CFC** algorithm. However, different from findings in the original publication, where $b = e - 1.7$, we found that $b = e - 1.0$ performs best for our problem.

Results

For the evaluations first the performance of all three classifiers was computed for the single-domain classification task (scenarios *NewsNews* and *BlogBlog*). These results indicate the maximum achievable performance for all classifiers for the cross-domain task. The experiments revealed that the **CFC** achieves an accuracy value of 0.95 in the single-domain task, equally good as the accuracy of the **SVM**. The **KNN** performs slightly worse with an accuracy of 0.93. Similar results were achieved for scenario *BlogBlog*, where **KNN**, **SVM**, and **CFC** perform all with accuracy of 0.94.

For scenario *NewsBlog*, the most interesting scenario, the performance of all three classifiers drops significantly. The **KNN** algorithm is best with accuracy of 0.80, the **CFC** algorithm performs with accuracy of 0.78, and the **SVM** with accuracy of 0.76. Additionally, we reduced this scenario *NewsBlog* to a binary classification task, taking only news articles and blog entries from the classes 'politics' and 'sports'. The results for

5.2 Experiment Set 1: Quality Assessment of Classifiers

the binary task versus the five classes problem are shown in table 5.8. Note, in this scenario, the accuracy dropped less from scenario *NewsNews* to scenario *NewsBlog* due to the lower complexity of the binary classification task.

Table 5.8: Accuracy, training and classification time for cross-domain scenario *NewsBlog* with 2 and 5 classes

	accuracy 2-classes	accuracy 5-classes	training time 5-classes	classification time 5-classes	total time
KNN	0.85	0.80	~7sec	~166sec	~173sec
SVM	0.82	0.76	~900sec	~24sec	~924sec
CFC	0.84	0.78	~10sec	~2sec	~12sec

For completeness of the experiments we also evaluated the classifiers on cross-domain scenario *BlogNews*. The **KNN** algorithm performs slightly better (accuracy of 0.83) than the **CFC** classifier with accuracy of 0.82. The **SVM** is worst with accuracy of 0.78. From the results of the conducted experiments, we can derive that in many cases, the **KNN** algorithm works slightly better than the **CFC** and the **SVM**. However, when regarding the computation time, the **CFC** is by far the best as can be seen in table 5.8.

We visually analyzed the classification results with the Class Radial Visualization. The visualizations for scenarios *NewsNews* and *NewsBlog* are depicted in figure 5.4. The three different classifiers result in three different shapes in the visualization. The **KNN** exhibits a polygonal shape, the **CFC** a star-like shape and the **SVM** a polygonal shape with emphasized diagonals. Further, **KNN** exhibits a discrete probability distribution. This can be derived from the fact that between two classes a maximum of 11 discrete confidence steps can occur, if majority weighting is applied. This also holds for the multi-class assignment.

In the visualization for the **CFC** classifier it can be seen that there are many items on the boundary between classes 'sports' and 'politics' as well as between 'politics' and 'economy'. Obviously the classifier has problems separating texts about politics from texts about sports and economy. The class 'sports' seem to be best separable. In the cross-domain case (figure 5.4e), the star-like shape is contracted compared to the single-domain case (figure 5.4b). This indicates that the classifier became more uncertain in the cross-domain task although the accuracy is still high.

From the visualization, it can be derived that the **SVM** has a clear tendency towards the category 'politics'. Also, the visualization gives the impression that the **SVM** analyses a set of binary classification problems since the test items are placed along the connecting lines between all classes. Also in the cross-domain case (figure 5.4f) the bias towards the class 'politics' remains visible and seems to be extended, since the rectangle for the class politics is filled more, i.e. the class attracts more items. (Note: this can be concluded because we have a nearly uniform distribution of classes in our data set).

5 Experiments

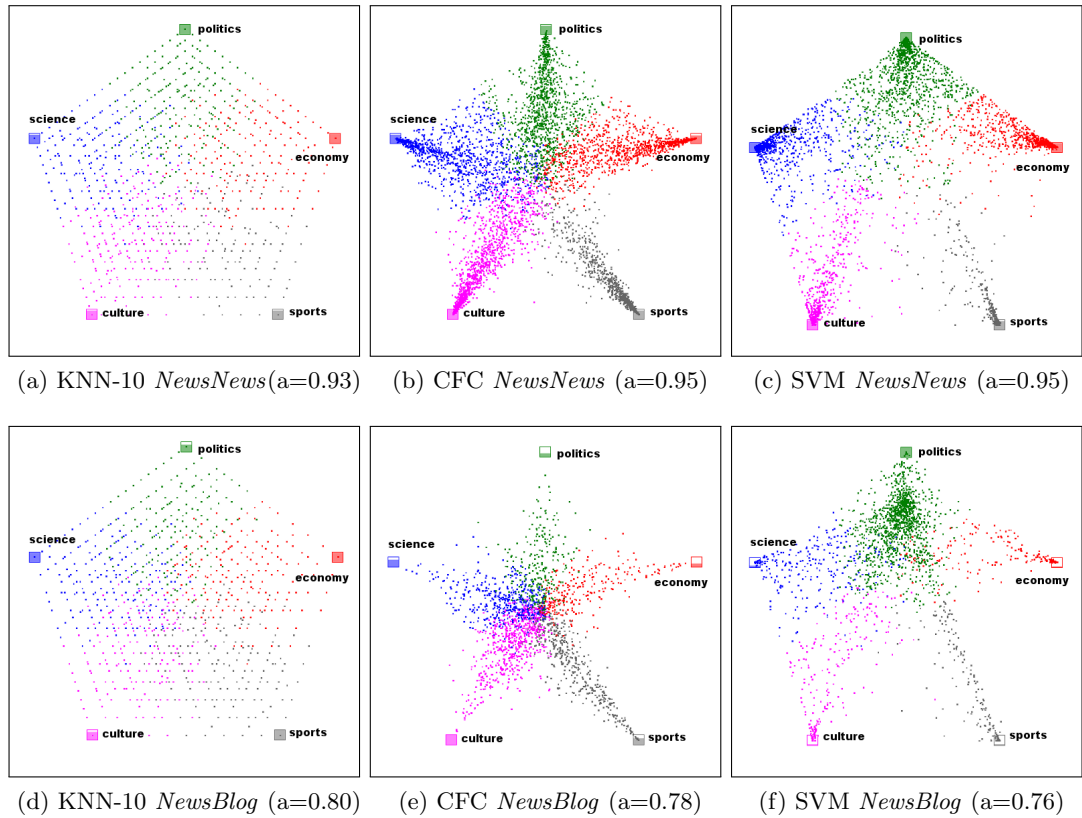


Figure 5.4: Visually comparing classifier visualization for different scenarios and different classifiers. 'a' denotes accuracy. KNN, CFC, and SVM classifier. *NewsNews* (first row) and cross-domain *NewsBlog* scenario (second row).

Additionally, we visually investigated the generalization abilities of the classifiers in the cross-domain setting using the Confusion Maps. The Confusion Maps are shown in figure 5.5 for scenarios *NewsNews* and *NewsBlog*. In the single-domain task (*NewsNews*) the Confusion Maps for the training and evaluation set are quite similar to each other indicating that all classifiers exhibit a good generalization behavior. In the cross-domain task (*NewsBlog*), the evaluation maps show darker colored off-diagonal elements whereas the Confusion Map for the training data reveals a distinct diagonal. From this, it can be derived that the classifiers overfit towards the training data in the cross-domain task, which means that their generalization abilities are lower than in the single-domain task.

Discussion

The **KNN** classifier is best in terms of accuracy but worst computation complexity for classification. The **CFC** classifier has second best accuracy in the cross-domain task. Its accuracy drops less than that of the **SVM** from the single-domain task to the cross-

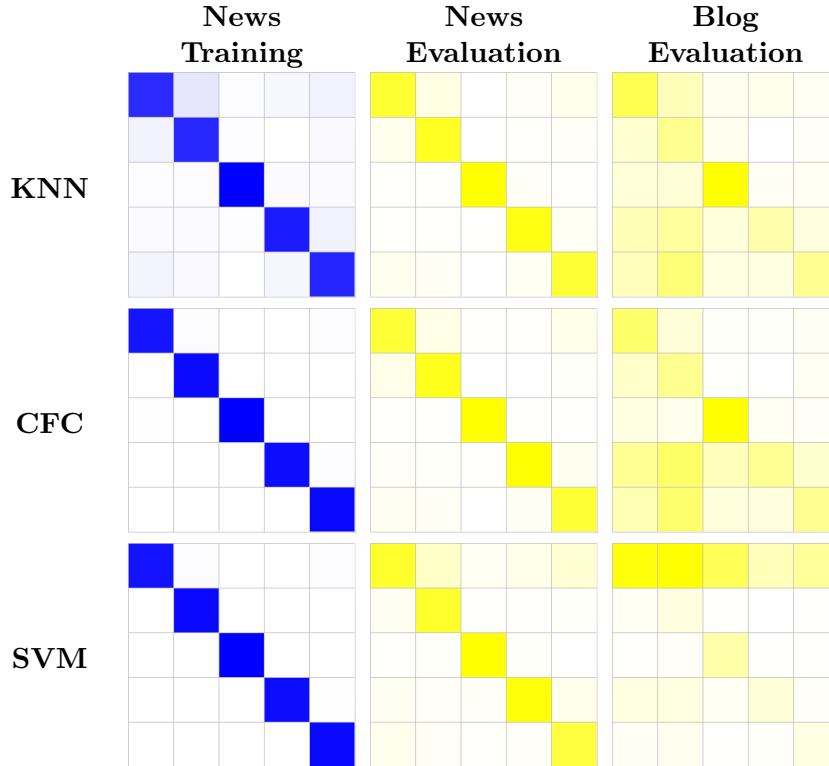


Figure 5.5: Comparing Confusion Maps for single-domain and cross-domain classification. First and second row correspond to scenario *NewsNews*, first and third row correspond to scenario *NewsBlog*.

domain task. Comparing the visualizations of **SVM** and **CFC**, the **CFC** places less test items on the outer boundaries, between the categories 'politics' and 'economy', as well as 'politics' and 'science'. The reason for this is that the centroid vectors overlap to a certain extent.

The visualization also reveals that the **CFC** does not prefer any class, in contrast to **SVM** ('politics'). This better reflects the a-priori probabilities that we train on an equally distributed corpus. For the *NewsBlog* scenario, all classifiers exhibit a very similar visual distribution compared to the single-domain task. That is why we expect that the algorithms perform similar on the cross-domain task. However, the correctness of the algorithms' decisions can only be verified when investigating the misclassifications (as depicted in figure 5.6). The misclassification of **KNN** is equally distributed as one can see in figure 5.6a. The **SVM** has several misclassifications in category 'science' with confidence nearly 1. This indicates that some of the support vectors cannot be generalized from the news domain to the blog domain.

The visual impression is reflected in the accuracy result. In contrary, the **CFC** has nearly no misclassification very close to the classes whereas the really misclassified items

5 Experiments

are placed more in the center of the visualization (region of low confidence). However, comparing the visualizations for the single-domain task and the cross-domain task, the shape changes most for the **CFC** classifier. It becomes more uncertain in its decisions for all items. This may give a hint, that if the vocabulary of the domains changes even more, the model of the **CFC** classifier may not be a good predictor any more.

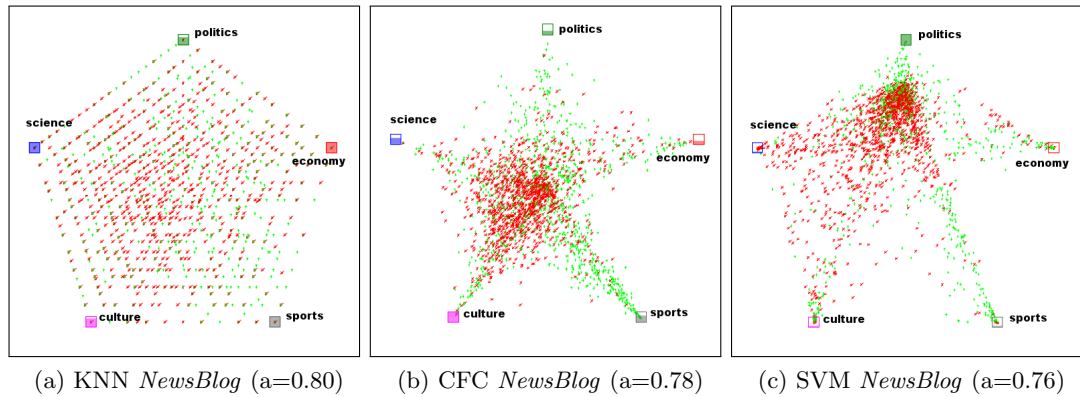


Figure 5.6: Visualizing misclassification of the classifiers KNN, CFC, and SVM for the cross-domain scenario (*NewsBlog*).

Summary

This experiment investigated whether the Class Radial Visualization and Confusion Maps may aid machine learning experts in understanding classifier behavior on a text data set. The chosen classification task is a cross-domain task. Several classifier-data set combinations were compared, by standard evaluation measures and further, using visualizations. It was shown that the visualizations allowed the machine learning experts to draw several additional conclusions – beyond the performance measures. An example is the bias towards special classes of the **SVM** classifier. Other aspects shown by the performance measures were reflected in the visualizations (e.g. generalization ability).

In one sentence the outcome of this experiment can be summarized as follows:

The Class Radial Visualization and Confusion Maps can help experts to understand arbitrary classification models for text classification tasks.

5.3 Experiment Set 2: Visual Active Learning

The question answered by this set of experiments on Visual Active Learning is:

Can user feedback on classification models through interactive visualizations, more specifically an interactive version of the Class Radial Visualization, be used to improve classification models? Is there a benefit over automatic methods?

The classical approach to active learning incorporates the user only to label examples previously automatically selected. In the classical active learning setting the procedure is as follows: (i) the learning algorithm is bootstrapped, (ii) the learning algorithm is applied on the evaluation data, (iii) the learning algorithm selects examples from the evaluation data whose labeling seems most beneficial in terms of a target function (i.e. classifier accuracy), (iv) the selected examples are labeled by an oracle (e.g., a user), (v) the learning algorithm updates its hypothesis by using these newly labeled examples. Steps (ii) to (v) are repeated until no more unlabeled examples exist, or a threshold for the target function is reached. The proposed user-centered approach to active learning alters step (iii), instead of letting the learning algorithm select the examples to label, the user herself selects these examples. Apparently, the user needs tools to estimate the benefit of selecting examples.

Two different experiments were performed to answer the question mentioned at the beginning, a single-user experiment and user simulations.

Single user experiment: In this experiment a user labels training items using the Class Radial Visualization. Using these labels, a classifier is trained. The measured accuracy of the trained classifier is compared to a classifier trained on randomly selected examples. The goal is to compare Visual Active Learning to random selection and investigate whether larger scale experiments may give further insights.

User simulations: In this experiment, Visual Active Learning is compared to classical active learning and the random baseline. Due to the huge number of tested classifiers and data sets the experiment uses models of user selection strategies. A shorter version of the user simulations described in section 5.3.2 is published in [SG10].

General Procedure: The general procedure of both, the single user experiment and the user simulations is essentially the same. The procedure is summarized in algorithm 3. The input for the procedure are a classifier, the set of classes C , initial training data set D_l , the test data set D_t , the evaluation data set D_e , and some selection strategy s . Note, that for the single user experiment, the test data set need not be labeled. On the contrary for the user simulations we need a fully labeled test data set.

5 Experiments

1. bootstrap classifier on $|C|$ randomly selected training examples
2. evaluate the classifier on D_e
3. **loop**
4. **for** $i = 1 \rightarrow |C|$ **do**
5. **if** s is a visual selection strategy **then**
6. visualize model
7. **end if**
8. select next item to label from D_u according to s
9. label selected item, add to D_l
10. retrain classifier on D_l
11. **end for**
12. evaluate the classifier on D_e
13. **end loop**
14. **return** collected performance measures for each step

Algorithm 3: Overall procedure of the experiments on Visual Active Learning

5.3.1 Single User Experiment

In this section two classifier learning scenarios are compared: random sampling and selection in the interactive visualization Class Radial Visualization by a user. This experiment is designed to investigate whether Visual Active Learning can outperform random sampling (it does not compare to classical active learning strategies, this is part of the second experiment).

More specifically, the question answered by this experiment is:

Can user feedback using the Class Radial Visualization be used to improve classification models? Is there a benefit over random sampling?

Procedure

We run the experiment on four different data sets and five classifiers with one user. The data set and classifier combinations were as follows: The Iris data set (see section 5.1.1) was trained with a [Mult-Layer Perceptron \(MLP\)](#) from the Weka Machine Learning library [[HFH⁺09](#)]. On the COIL-20 data set (see section 5.1.2) we run a [SVM](#) classifier from the LibSVM library [[CL01](#)] and a [KNN](#) classifier using our own implementation. On the R8 subset of the Reuters2178 text data set (see section 5.1.3) a linear [SVM](#) from the LibLinear library [[FCH⁺08](#)] was trained. The the R8 subset of the Reuters2178 text collection is abbreviated with REU-R8 in the experiments. On the second text data set, the APA Topics (see section 5.1.4) our own implementation of the [CFC](#) classifier was applied. For the text data sets we used a noun vector space, and [TF-IDF](#) weighting. We chose to use $k = 7$ for the [KNN](#) classifier on the COIL-20 data set because [[SHZ⁺07](#)] reported $k = 7$ to perform best on this data set.

All data sets were fully labeled. The classifier was bootstrapped with c (c =number of classes) randomly selected items for both conditions. Subsequently, c samples were selected either by random choice or by the user and the classifier was retrained on these samples and evaluated on the test set (batch-mode). The accuracy of the classifier on the test set was recorded. The employed user strategy was as follows: the user does not label items that seem to be already well classified, she aims at the seemingly uncertain items first (items in the center of the visualization). For the combinations Iris data set with **MLP**, and COIL-20 data set with **SVM** classifier the user does not label correctly classified items, i.e. if such an item is selected it is not labeled.

Results

In this section the results for both, random sampling and visual active learning for all data set and classifier combinations are reported.

Iris data set with MLP: Table 5.9 and figure 5.8a show the results for the **MLP** on the Iris data set. Note, that the user strategy for this data set and classifier combination was to only label misclassified examples. The accuracy of the bootstrapped classifier (trained on 3 items) is 69%. After the first training step the accuracy for the user-trained classifier heavily drops to 57% whereas the classifier trained with random sampling reaches an accuracy value of 94% on the evaluation data. In every subsequent step, the visual classifier performs better than the classifier trained on random samples. After the third step (10 items) no more misclassified items are present in the test data set, still the accuracy increases when more labeled items are added to the training set. After 18 training samples, the classifier obviously does not get any additional information, the accuracy value remains stationary in both conditions: 96% for the visual classifier and 90% for the classifier trained on randomly selected samples.

Table 5.9: **MLP** on Iris data set: accuracy on evaluation data for random sampling and visual sampling, * means, no more misclassified items in test data

#items	3	6	9	12	15	18	21	24
random	69.4	93.9	89.8	73.5	89.8	89.8	89.8	89.8
user	69.4	57.1	95.9	91.8*	93.9	95.9	95.9	95.9

COIL-20 data set with KNN and SVM classifiers: Table 5.10 and figure 5.8c show the results for the **KNN** classifier on the COIL-20 data set. The accuracy after bootstrapping the classifier is 8.6%, which is slightly better than the accuracy of the trivial classifier (trivial accuracy = 5%). Subsequently, user sampling always outperforms random sampling in each step. After 200 trained samples the accuracy of the classifier trained with random samples from the data set is 53.4%. The classifier trained by the user performs better, yielding an accuracy value of 58.6% after 200 training samples.

5 Experiments

Table 5.11 and figure 5.8b show the results for the SVM classifier on the COIL-20 data set. Note, that the user strategy for this data set and classifier combination was to only label misclassified examples. The accuracy after bootstrapping the classifier is 2.7%, which is worse than the accuracy of the trivial classifier (trivial accuracy = 5%) and the KNN classifier with the same training samples. Subsequently, user sampling sometimes outperforms random sampling and vice versa until the classifier has seen 100 training samples. After 120 training samples, random sampling is always outperformed. After 200 trained samples the accuracy of the classifier trained with random samples from the data set is 63.5%. The classifier trained by the user performs better, yielding an accuracy value of 73.5% after 200 training samples, which is by far better than the trained KNN classifier.

Table 5.10: COIL-20, PCA features, KNN, Comparing accuracy values on evaluation item set for random sampling and user selection

#items	20	40	60	80	100	120	140	160	180	200
random	8.6	12.4	20.9	28.9	33.8	41.4	46.4	49.4	52.1	53.4
user	8.6	16.5	30.2	33.3	36.9	43.9	46.8	53.4	56.8	58.6

Table 5.11: COIL-20, PCA features, SVM. Comparing accuracy values on evaluation item set for random sampling and user selection

#items	20	40	60	80	100	120	140	160	180	200
random	2.7	7.4	11.8	19.8	31.0	35.7	44.1	52.2	56.5	63.5
user	1.9	1.7	7.0	25.7	26.6	38.2	61.0	62.7	69.8	73.0

REU-R8 with SVM classifier: In table 5.12 and figure 5.8d the results of the SVM classifier on the Reuters data set are reported. The accuracy after bootstrapping the classifier is 50.4%. Until 32 items were selected for training, random sampling always outperforms user sampling. In subsequent steps, however, the behavior changes, and user sampling outperforms random sampling. After 88 trained samples the accuracy of the classifier trained with random samples from the data set is 76.2%. The classifier trained by the user performs better, yielding an accuracy value of 83.9% after 200 training samples.

Table 5.12: SVM classifier on REU-R8 data set: accuracy on evaluation data for random sampling and user selection

#items	8	16	24	32	40	48	56	64	72	80	88
random	50.4	56.4	59.8	59.3	62.3	65.3	67.8	71.6	72.3	72.8	76.2
user	50.4	55.5	57.9	64.6	74.4	79.4	80.9	81.3	82.7	85.0	83.9

APA data set with CFC classifier: Table 5.13 and figure 5.8e show results for the CFC classifier on the APA data set. The accuracy of the classifiers after bootstrapping with 5 items is 32%, 12% higher than the trivial accuracy of 20% (5 classes, evaluation data nearly uniformly distributed over the classes). Up to the last simulated step, the accuracy of the visual classifier is higher than that of the classifier trained with randomly selected items. The difference ranges from 1.1% (95 items) to 17% (20 items).

Table 5.13: CFC on APA data set: accuracy on evaluation data for random sampling and user selection

#items	5	10	15	20	25	30	35	40	45	50
random	32.5	38.5	40.2	39.4	45.2	47.9	48.8	51.0	49.8	52.5
user	32.5	43.8	54.4	56.8	57.4	60.5	63.2	64.1	65.3	67.1
#items	55	60	65	70	75	80	85	90	95	100
random	59.5	61.9	62.1	63.2	63.4	65.5	67.3	70.0	70.5	69.8
user	67.8	68.9	68.8	69.6	68.3	68.3	69.7	71.4	71.6	71.9

The accuracy curve for the visual classifier (compare figure 5.8e) is steeper at the beginning, the slope decreases after 15 items, whereas the slope of the accuracy curve of the classifier trained on random samples has a steadily increasing slope. In this scenario the classifier largely benefits from the user selection. The visualization of the test item set after having trained 100 items is shown in figure 5.7. For the classifier trained with random samples it can be seen that the class 'science' has less items (confidently) assigned than the classifier trained on user selected items. This means, this class seems to be not well discriminated from at least one other class. This fact is confirmed by the class confusion matrix on the right-hand side of figure 5.7b: The class 'science' can not well be distinguished from the class 'culture'.

The accuracy in the uncertainty sampling case (compare figure 5.8e) shows a steeper curve at the beginning than the curve for random sampling. E.g., with as little as 20 training samples in the uncertainty sampling case we already reach an accuracy of 56.8%, a value that is reached in the random case with approximately 50 items (accuracy = 52.5%). Further, with user sampling the curve shows an asymptotic behavior, whereas in the random case the curve is more linear with local minima. Although the accuracy of the classifier after being trained on 100 items is nearly equal in both cases (69.8% for random, 71.9% for uncertainty) in the uncertainty case it seems more clear that a step is reached where only marginal changes in the accuracy are to be expected. Remaining data items would only contribute to a much lesser extend, than the previously trained ones. This is not the case in the random case, each item approximately contributes equally to the classifier's accuracy (linear curve). If one needs to make a decision when to stop the (probably costly) training process, this decision is better supported in the latter case.

Figure 5.8 shows the accuracy plots comparing Visual Active Learning and random

5 Experiments

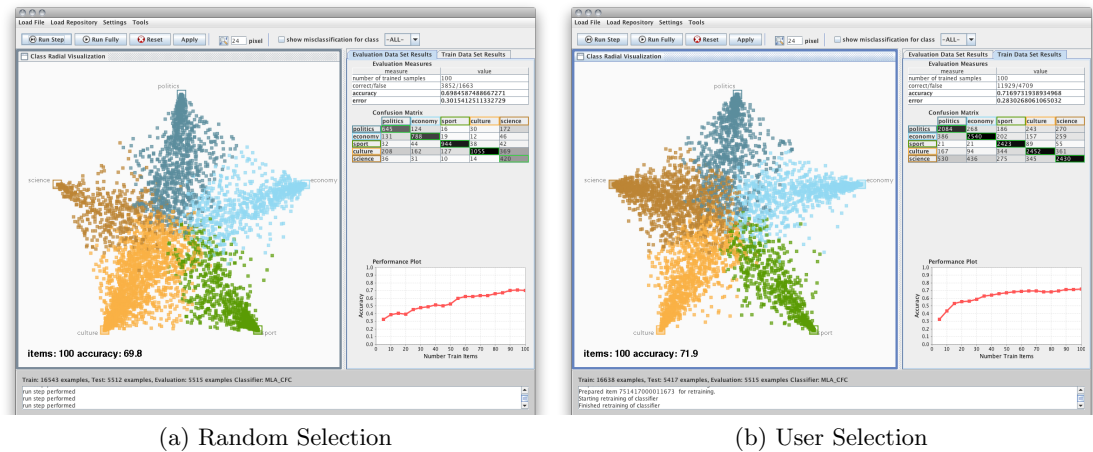


Figure 5.7: Screenshot classifier visualizations: CFC APA *NewsNews*, random selection and user selection.

sampling for all data set and classifier combinations. It can be seen, that after enough training samples were provided Visual Active Learning always outperforms random sampling. However the margin differs for each data set and classifier combination.

Discussion

The results of this experiment show, that in almost all data set and classifiers combinations Visual Active Learning outperforms random sampling. This fact is summarized in table 5.14. Further, it seems to make no difference whether the user selects uncertain examples or uncertain examples that are misclassified. However, since only one trail was performed, it is not clear how well the finding generalize. The experiments in the next section aim at providing more statistical evidence in this regard.

Table 5.14: Overview of the results comparing random selection and user selection. $R \ll U$ means, random is worse than visual selection in terms of accuracy. $R == U$ means, no difference. (* means that the user selected only misclassified examples)

Data Set	Classifier	
Iris*	Multilayer Perceptron	$R \ll U$
COIL-20*	SVM	$R == U$
COIL-20	KNN	$R \ll U$
REU-R8	SVM	$R \ll U$
APA	CFC	$R \ll U$

5.3 Experiment Set 2: Visual Active Learning

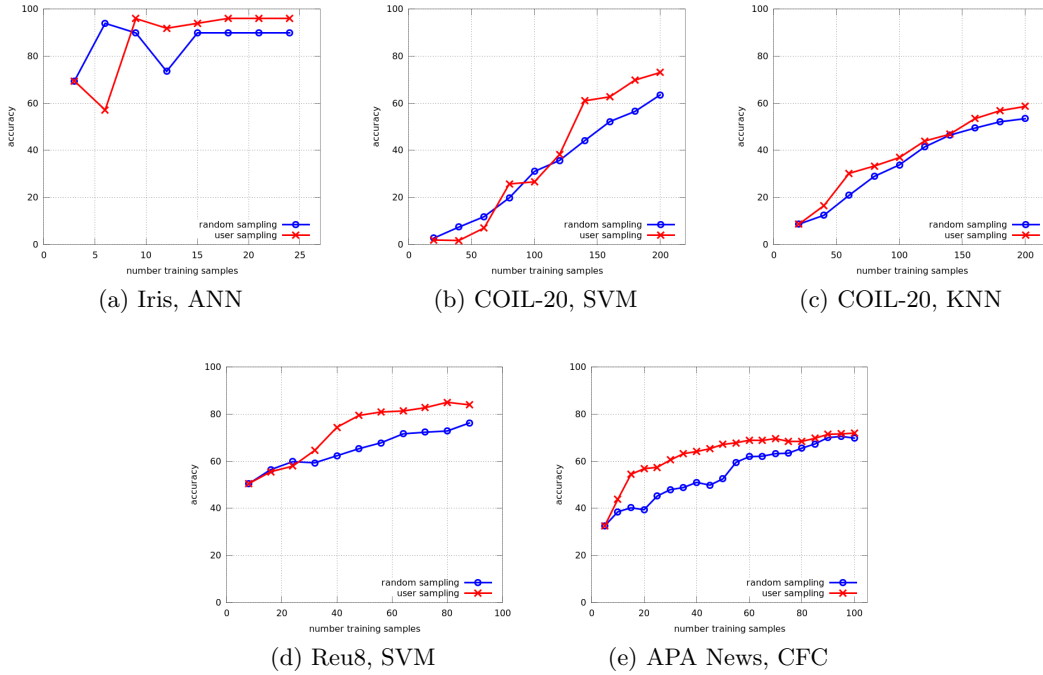


Figure 5.8: Accuracy plots of all data set and classifier combinations. For random sampling and Visual Active Learning.

Summary

This experiment considered the adaptation and construction part of the research question. It was investigated whether a user-centered approach to active learning can outperform naive learning strategies. More specifically, it was tested, whether Visual Active Learning using the Class Radial Visualization outperforms random sampling. The user was trained in the usage of the Class Radial Visualization and therefore, knew where to search for potential, new training items. The experiment was performed on four different data sets and five classifiers. The results show, that in almost all cases Visual Active Learning outperforms random sampling, in one case there was no significant difference. Also, the results suggest, that it makes no difference in terms of classifier performance if the user only selects misclassified examples (instead of all uncertain examples). This latter fact and whether Visual Active Learning also outperforms classical Active Learning will be investigated in the next experiment (see section 5.3.2).

In one sentence the outcome of this experiment can be summarized as follows:

Visual Active Learning with the Class Radial Visualization outperforms random sampling in nearly all tested conditions and never performs worse.

5.3.2 User Simulations

This section describes performed user simulations to investigate whether it is beneficial to not only let the user label the example, but also let the user select the example to label. The means to help the user to investigate, assess and select examples is the visualization described in section 4.1. This setting is more general than the experiment described in the previous section.

In section 5.3.1 we found out that letting the user label examples with the visualization lead to an increased accuracy of the classifier in the range of a small number of samples. However, these experiments were performed by a single user in one single trial and can not be generalized. This section describes experiments that try to model user behavior in order to generate a sufficient number of trials to allow for statistical analysis. If the user is modeled appropriately a large number of labeling experiments could be run automatically allowing to test the hypotheses without time-consuming user experiments.

On order to make a rather general conclusions a variety of combinations must be tested. The independent variables for this experiments are (i) the selection mode, (ii) the classifier (with standard parameter settings), and (iii) the data set. The outcome of the experiment could depend on a certain choice for either of them. Thus, for the experiment, we chose four data sets and three classifiers.

The huge amount of simulations could not have been done with real users (The complexity of the evaluation: 4 data sets, 3 classifier, 2 user selection and 2 user labeling strategies, 10 runs for each setting. $4 \times 3 \times 2 \times 2 \times 10 = 480$ user. 100 samples to label on average. Estimated user labeling time for on sample: 3 s for image data sets, 30 s for text data set per example. This means ≈ 50 min per user for one text data set.). Instead the obtained statistical results show a direction for user experiments and allow an estimate what to expect in real user studies. In turn, we plan to derive new visual selection models from the user studies which then can be statistically evaluated using the simulation framework.

The question answered by this experiment is:

Can user feedback using the Class Radial Visualization be used to improve classification models? Is there a benefit over random sampling and classical active learning?

Hypotheses

The hypothesis for this experiment is that Visual Active Learning with trained users outperforms random selection but in turn is outperformed by classical active learning. Further, we expect an accuracy increase of the classifier if we only use misclassified examples as further training examples. One can argue that using only misclassified samples is a rather artificial setting, because in practice one does not have this infor-

mation. However, since we include the user in the selection process this information will be available at the time of the selection (since the user can decide whether a given item is misclassified). In short, the hypotheses we investigate in this experiment are the following:

- H1** Visual active learning outperforms greedy active learning.
- H2** Visual active learning is outperformed by random sampling.
- H3** Labeling only misclassified (uncertain) examples significantly improves classifier accuracy compared to labeling all uncertain examples.

One may argue that there is not much benefit if visual active learning outperforms greedy active learning. Greedy active learning has been outperformed by other active learning strategies [Set10] – such as SVM-specific strategies [TK01]. And thus we would not compare our approach to state-of-the art. However, greedy active learning is state-of-the art for classifier-agnostic active learning and our proposed strategy is independent of specific classifiers, too.

Procedure

Figure 5.9 depicts an overview of the experimental procedure. From observations in the single-user experiment and knowledge about the layout principles of the underlying visualization two selection models ('gaussian' and 'convhull') were derived. They are probabilistic descriptions of a user's selection behavior. Furthermore, a clear benefit of Visual Active Learning could be that users may choose to label only misclassified examples – not the ones that are uncertain and correct. This decision about a misclassification can not be done by any automatic procedure, it must be done by humans. Thus, we implemented two different labeling strategies to verify their influence on the classifier performance ('all' and 'misses').

In order to compare with classical active learning, two classifier-agnostic active learning strategies ('minconf' and 'entropy') were chosen. For details about this selection models see equations 2.33 and 2.34 in section 2.5. Finally we compare the two active learning selection strategies and the visual active learning strategies to the random baseline ('random') in our user simulations.

The tested combinations of classifiers and data sets are summarized in table 5.15. The used classifier implementations are the same as in the single user experiment: SVM from the LibSVM library [CL01] and the LibLinear library [FCH⁺08] and our own implementation of the CFC and KNN classifier.

The user selection models and classical active learning selection models compared in the evaluation are summarized in table 5.16.

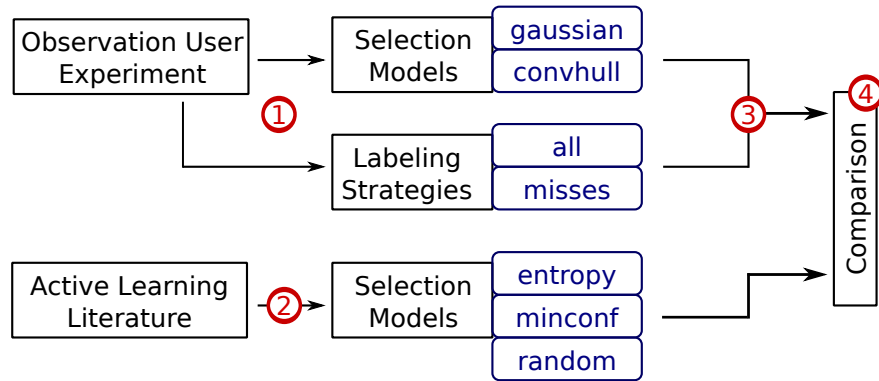


Figure 5.9: Overview of the experimental procedure: ① In the user experiment in section 5.3.1 two selection models and two labeling strategies were observed and mathematically modeled. ② Two selection models were taken from the active learning literature, the ‘random’ model serves as baseline. ③ The selection models and the labeling strategies from the user experiment are combined. ④ Visual Active Learning strategies are compared to classical active learning on different data sets and different classifiers.

Evaluation approach For each of the classifier and data set combinations five different selection modes are compared: random selection, entropy-based selection, minimum confidence selection, and two selection modes that model user behavior. The selection modes are described in more detail in the following. For all selection modes the classifier was bootstrapped with c randomly chosen samples ($c = \text{number of classes}$). The learner was trained in serial-mode, i.e., one example at a time was labeled and added to the training set. Afterwards, the model was retrained and the test data set reclassified. After every c examples we evaluated the classification error on the evaluation data set. To avoid random biases due to initialization or within the different selection strategies, we run every experiment ten times and averaged the results.

Models of User Selection Strategies In the first selection model, ‘*gaussian*’, we simulate user selecting examples mostly from the center of the visualization. This selection model assumes that the user tries to select the most ambiguous samples, i.e. those samples classes are mostly competing for. These samples are placed in the center of the visualization. To simulate this behavior we select an example with probability according to a bivariate Gaussian distribution over the center of the visualization. This means, the probability $p(x, y)$ to select an example at position (x, y) is proportional to $p(x, y) \approx \mathcal{N}_2(\mu_1, \mu_2, \Sigma)$ with $\mu_1 = \mu_2 = (0.5s, 0.5s)$, $\Sigma = \text{diag}(0.1s)$ and s being the diameter of the circle. However, the center does not contain the most ambiguous examples in all cases. For example, if during active learning only two classes have assigned training examples, the remaining test examples are distributed along the line connecting both classes which not necessarily runs through the center.

Table 5.15: Overview of compared data set and classifier combinations

Data Set	Classifier	Corresponding single-user experiment
COIL-20	KNN	see section 5.3.1
	SVM	see section 5.3.1
20NEWS	CFC	
	SVM	
APA	CFC	see section 5.3.1
	SVM	
REU-R8	CFC	
	SVM	see section 5.3.1

Table 5.16: Overview of compared selection strategies and user labeling strategies

Strategy Type	Strategies
baseline	random
active learning	entropy minconf
visual active learning	gaussian-all gaussian-misses convhull-all convhull-misses

The second selection model *'convhull'* simulates user who do not pick from the center of the visualization but rather judge the distribution of the samples and then select the presumably most ambiguous sample based on this distribution. In particular, the "convhull" selection model calculates the convex hull around the visible unlabeled examples and centers a bivariate Gaussian distribution around the point with the smallest distance to the center of the circle. If the center of the visualization is inside the convex hull then the center of the visualization is taken as the central point of the Gaussian; hence, the "convhull" selection model becomes the 'gaussian' selection model described above. The covariance matrix for the 'convhull' model is set the same as in the 'gaussian' model. The convex hull is calculated using Graham's algorithm [Gra72]. An example for both user selection models, 'convhull' and 'gaussian', is shown in figure 5.10.

Of course there are other possible selection behaviors. For example, the user can select items uniformly across the visualization. We do not consider this selection model because it is equivalent to the random selection and we assume users familiar with the concepts of the visualization. We also not consider changes in user behavior for reasons of simplicity.

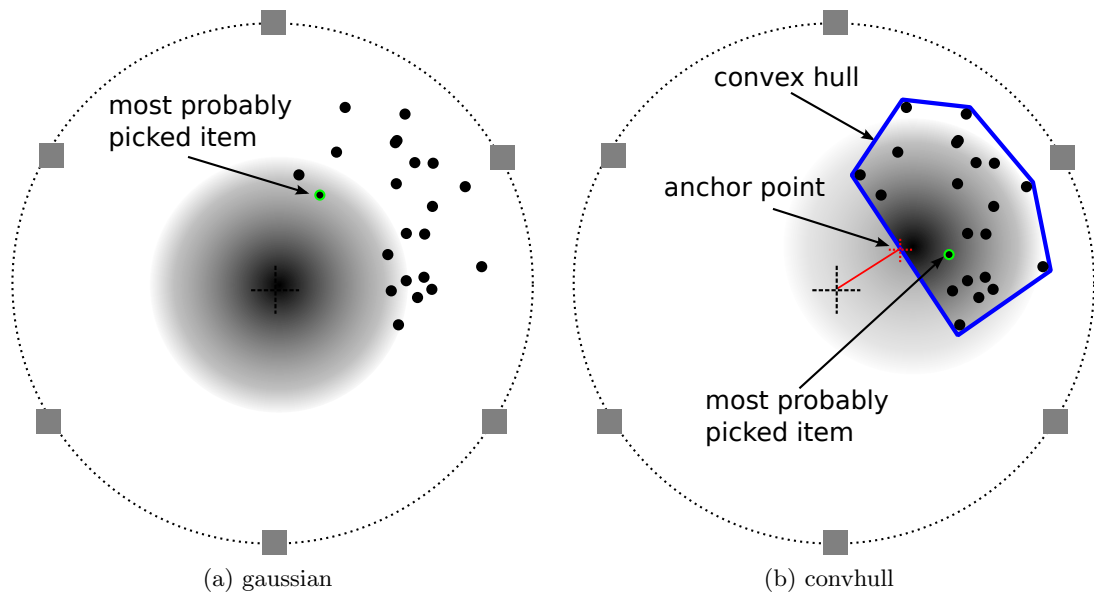


Figure 5.10: Models of user selection strategies

Models of User Labeling Strategies Independent of the selection model – which items to select next for investigation, the user may have different labeling strategies – whether or not to label the selected example. For the experiments two different strategies will be investigated. The first strategy is the so-called 'all' strategy: independent whether the item under investigation is correctly classified or not, the user labels it. The second considered labeling strategy is "misses" – which means the user only labels items that are wrongly assigned by the classifier. This latter strategy means that the user investigates items one after another and after she found a misclassification she would correct the label.

There are two reasons why only following the "misses" strategy would not work in the experiments. First, there might occur situations where no more misclassified items occur in the visualization (the classifier perfectly classifies the test data set). Second, no user would have the patience to investigate dozens of items before eventually being allowed to label one. Therefore, the "misses" strategy uses a threshold to limit the number of unsuccessful subsequent investigations. If this threshold is reached the item is labeled anew, independent of whether it was misclassified or not.

Results for Testing Hypothesis H1 and H2 (selection strategies)

Most noteworthy, there is no clear winning active learning strategy over all data sets. Moreover, we can not confirm that the compared uncertainty based sampling strategies have a clear advantage over random sampling.

5.3 Experiment Set 2: Visual Active Learning

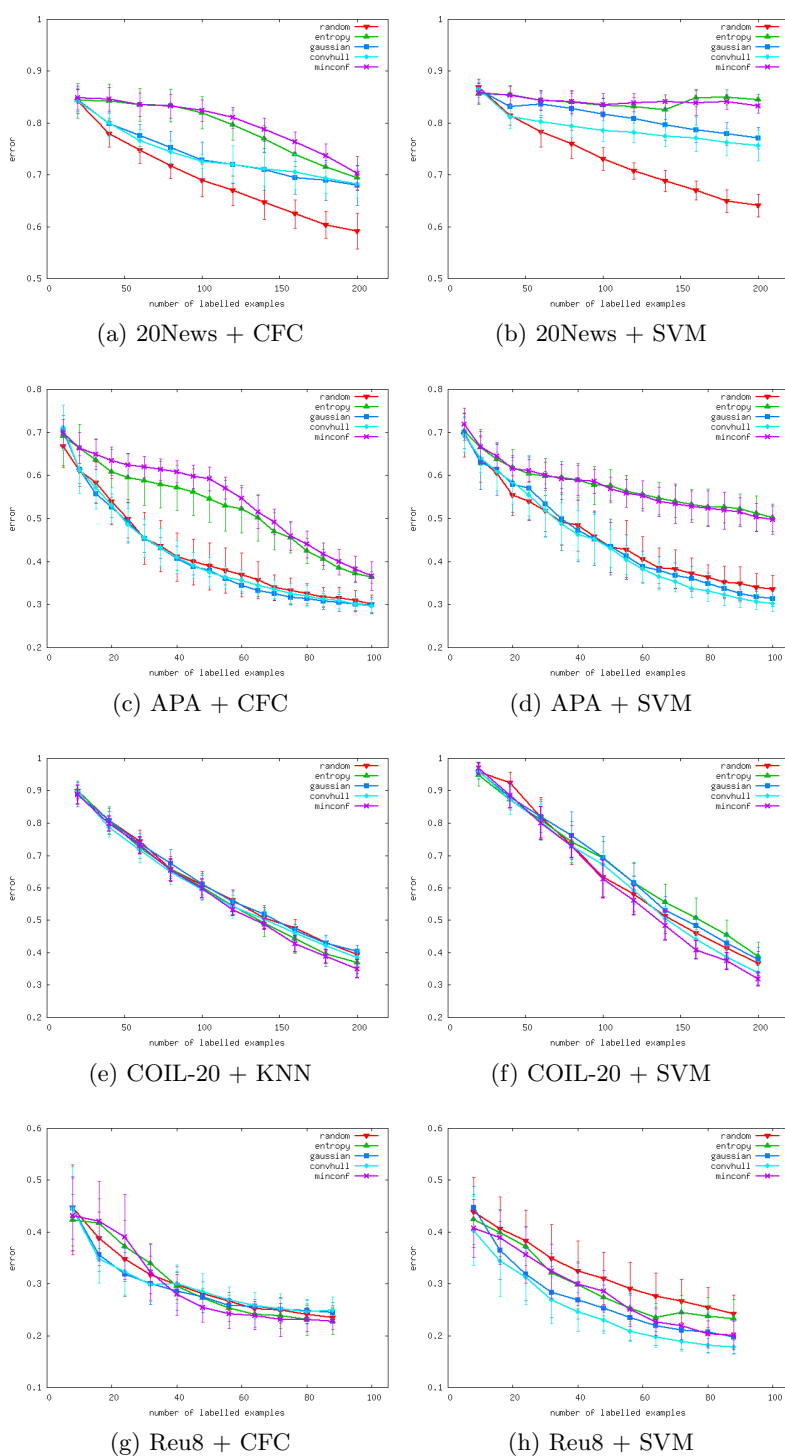


Figure 5.11: Comparing selection strategies for all data set and classifier combinations: Error rates averaged over 10 runs, showing mean and standard deviation.

5 Experiments

In particular, on the APA data set (see figure 5.11c for CFC), both non-visual learning strategies, 'entropy' and 'minconf', have not been able to improve the classification accuracy. The random baseline outperforms 'entropy' and 'minconf' by a large margin. The visual active learning strategies tend to be slightly worse at the beginning compared to the random baseline but could beat the random baseline by a significant fraction at the end.

On the REU-R8 data set the classifiers shows quite different behaviors (see figure 5.11h for SVM). The CFC seems to converge faster, since after 20 examples the classification error drops below 0.3 independent of the selection strategy. Moreover the standard deviation significantly decreases after 30 examples for all selection strategies, indicating a more stable behavior. The SVM classifier shows a different behavior. The random baseline is outperformed by all other strategies by a large margin. Moreover, the classical active learning strategies are outperformed by the user selection strategies, while the 'convhull' strategy performs best.

The COIL-20 data set is the only data set where classical active learning strategies outperform the visual (see figure 5.11f). In particular 'minconf' outperforms the random baseline in case of the SVM classifier while entropy-selection outperforms the random baseline in case of the KNN classifier (see figure 5.11e). Our visual strategies, especially the 'convhull' strategy performs quite similar to the random baseline.

The 20NEWS data set shows a special behavior (see figure 5.11b for SVM). All active learning strategies got beaten by the random baseline by a large margin of nearly 20%. However, visual active learning strategies perform better than classical ones.

On all data sets, classical active learning approaches have a smaller standard deviation over the runs than our visual approaches. However, on most data sets our visual methods perform still better taking worst-case runs, i.e. those with the highest standard deviation, into account.

From these results we can draw the following conclusions: (i) Our visually inspired active learning strategies outperform classical uncertainty based sampling strategies in most cases (20NEWS, REU-R8 and APA). (ii) We can confirm the findings summarized in [Set10], that the best sampling strategy depends on the application, the data set and the classifier and that the random baseline is not always outperformed.

Results for Testing Hypothesis H3 (labeling strategies)

Figure 5.12 summarizes the results of the experiments varying the labeling strategy ('all' and 'misses').

The results are partly surprising. We expected to see a clear gain in accuracy when comparing the standard 'all' strategies to the respective 'misses' strategies. The hypothesis of improving the accuracy when applying the 'misses' strategy can for instance be confirmed for REU-R8 with SVM "gaussian" (see figure 5.12h). In this case labeling only

5.3 Experiment Set 2: Visual Active Learning

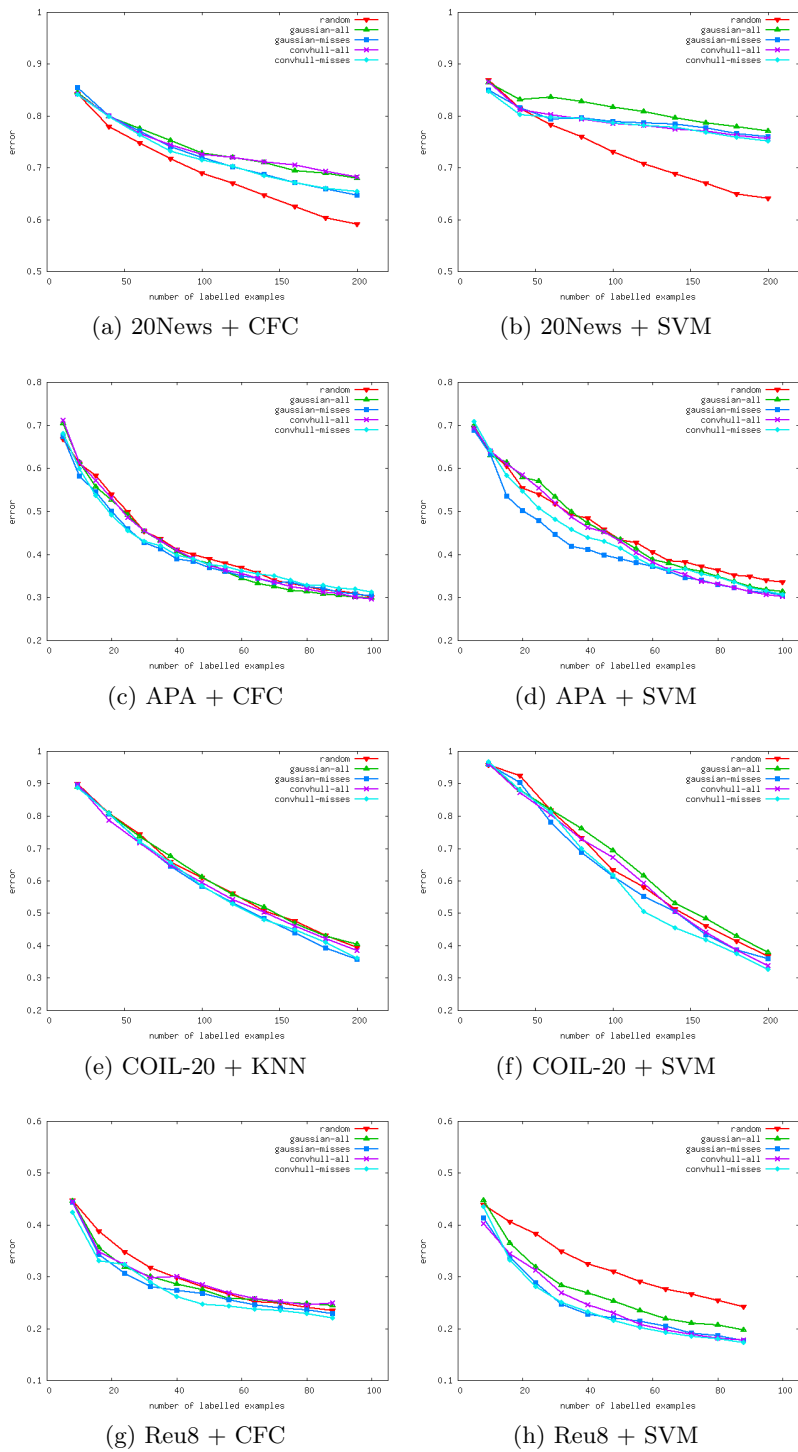


Figure 5.12: Comparing labeling strategies for all data set and classifier combinations: Error rates averaged over 10 runs, showing mean and standard deviation.

5 Experiments

misclassified examples leads to a significant improvement of the classifier accuracy. On the contrary, there is no effect on the accuracy when changing the 'all' to the 'misses' labeling strategy, for example in the setting APA and CFC using 'gaussian' selection (see figure 5.12c). Table 5.17 summarizes the effect of applying the "misses" strategies for all combinations of data sets and classifiers.

One possible explanation for the above described behavior is the following: The COIL-20 data set contains no noise and no outliers. The 'misses' strategy on this data set labels data items which are no outliers, are uncertain and misclassified and thus provides more (correct) information to the classification model than the 'all' strategy which leads to improved accuracy. On the contrary, the text data sets (REU-R8, APA, 20NEWS) contain outliers. Applying the 'misses' strategy could result in selecting outliers or noisy data items. The similar behavior of 'misses' and 'all' for the CFC algorithm could be explained by the properties of the CFC algorithm: The influence of one example on the class centroid is relatively small, the CFC regularizes well. Thus, neither the additional information of (uncertain and misclassified) nor selection of outliers have strong influence on the accuracy. In case of the SVM classifier (hard margin), outlier selection can result in a strong change of the hyperplane and thus in loss of generalization ability.

Table 5.17: Effect of labeling only misclassified examples. '+' : labeling only misclassified examples increases accuracy, 'o': no difference between labeling all or only misclassified examples

		gaussian	convhull
20NEWS	CFC	+	+
	SVM	+	o
REU-R8	SVM and CFC	+	+
APA	SVM	+	o
	CFC	o	o
COIL-20	SVM and KNN	+	+

Summary

With our experiments we showed that trained users would perform better than greedy active learning strategies in all cases and at least not perform worse than the random strategy while given more choice in the active learning process. The experiments for the different labeling strategies showed that the benefit of these strategies depends on the data set and the classifier.

The last finding points towards a crucial property in active learning: users have to be able to find out whether the chosen active learning strategy works or not and to switch the active learning strategy as needed. The selection using the interactive visualization

5.3 Experiment Set 2: Visual Active Learning

allows such switching of active learning strategies. Although, the "misses" strategy performs never worse than labeling all selected examples, there is no huge effect on the classifier accuracy in all tested conditions. This means, using the human knowledge about correctness of current decisions does not always improve the efficiency of a trained classifier. Therefore, for practical applications users do not have to be trained to focus on misclassified examples.

From conducted experiments and user simulations we conclude, that utilizing humans to only label examples in active learning settings is suboptimal. Giving users a more active role in terms of a visual selection of examples and in adapting their labeling strategies on top of tailored visualization techniques could increase labeling efficiency.

In one sentence the outcome of this experiment can be summarized as follows:

Visual Active Learning with the Class Radial Visualization outperforms classical active learning in all tested conditions and performs never worse than random sampling.

5.4 Experiment 3: Document Representation for Efficient Labeling

This section describes the experiment designed to answer the question whether the bottleneck identified in the experiments in section 5.5 and 5.3 can be overcome. Both previous experiments dealt with text classification, one with classifier construction alone, and the other with classifier construction and adaptation. In both cases, new training data has to be generated. Also, classified text documents have to be assessed whether they are correctly classified. For text classification this means, users have to read the texts in order to make the assessment and/or assign the correct label. It is not surprising, that the time needed to comprehend the texts is by far longer than the time required for the actual labeling (which can easily be implemented by drag and drop).

Thus, this experiment was designed to investigate, whether alternative representations of text (other than the obvious full-text) may support users in finding the category faster. It is crucial that these alternative representations can be generated automatically, otherwise the gain in speed would be nullified. More specifically, the question that will be answered by the experiment in this section is

What are good representations of the data to classify, more specifically of text documents, to speed-up the manual labeling process?

The experiment comprises of two steps:

1. Identification and application of automatic text representation forms.
2. Comparative user study to assess the possible speed up by using any of these text representation forms compared to the full-text representation. Furthermore, it is assessed whether the labeling accuracies change when using different text representation forms.

The experiment presented in this section has been published in [SUG11].

5.4.1 Procedure

This section presents the methodology to evaluate the effect of different text representations on manual labeling speed and accuracy.

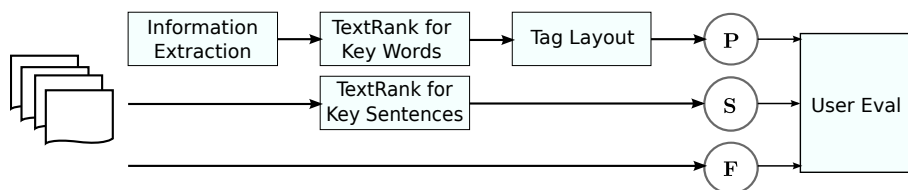


Figure 5.13: Overview of the evaluation methodology

Figure 5.13 gives an overview of our methodology. Starting from text documents (on the left) three different paths for generating the three different text representation forms are shown. In this paper we use the word 'condition' as a synonym for the text representation form, because each text representation form resembles a condition in our user evaluation. The three different conditions are denoted as **F** for full-text, **S** for key sentences (and named entities) and **P** for key phrases. In the following subsections the steps to generate the key phrases and key sentences are explained in detail. The full-text conditions serves as baseline to which we compare the users' labeling accuracy.

Keyword and Key Sentence Extraction

We applied the TextRank algorithm [MT04] to extract key sentences and key words from a document. The TextRank algorithm is a graph-based ranking algorithm. The relevance of a node in the graph is determined by a voting mechanism. All predecessor nodes vote for a specific node, the score of a node is calculated from the scores of its predecessors. The final score for all nodes is determined by iteratively calculating the score for each node until the algorithm converges. To apply the TextRank algorithm, the documents need to be preprocessed. For preprocessing we used a standard information extraction pipeline consisting of the following steps: tokenization, stemming, stop-word removal, [Part-of-speech \(POS\)](#)-tagging and named entity extraction. The named entities of type 'person' were added to the extracted key phrases and together they represent the key phrase condition **P** in the experiments.

TextRank for Key Sentence Extraction: For extracting key sentences the graph is constructed as follows: One node is created for each sentence. An edge between two nodes is created if their sentences are similar to each other. The similarity between two sentences is a function of their overlapping words, for instance the cosine similarity of the feature vectors of the sentences in a vector-space representation. On this weighted, undirected graph the graph-based ranking algorithm is applied. After the algorithm has converged, the nodes are sorted according to their score and the topmost nodes are selected.

TextRank for Keyword Extraction: For extracting keywords the graph is constructed as follows: (i) the text is split into tokens, (ii) [POS](#)-tags are assigned to each token, (iii) for each token or all tokens for a specific [POS](#)-tag a node is created, (iv) a link between two nodes is created if the words co-occur within a given window. On this unweighted, undirected graph, the graph-based ranking algorithm is applied. After the algorithm has converged, the nodes are sorted according to their score and the top T words are taken for post-processing. In the post-processing step, sequences of adjacent keywords are collapsed to multi-word keywords also termed key phrases.

The experiments used a slightly adapted version of the TextRank algorithm, namely, only one graph was constructed containing both, words and sentences.

Keyword Layout

The key phrases extracted by the TextRank algorithm may originate from any location of the source text. Two key phrases may belong to the same sentence and share the same context but they also may not. Consequently two key phrases have a relation as they are extracted from the same text but we do not know (anymore) which relation it is. We chose to use a layout for the key phrases and named entities that reflects this uncertainty in the relations. A line-by-line (Western reading-direction) layout would indicate either a relation in reading direction between the words, or none relation at all for people used to read tag clouds. We chose a layout algorithm from the family of tag layout algorithms described in [SKK⁺08], where the words are laid out in a circular manner, starting from the center-of-mass of the visualization boundary. The interesting property of this layout algorithm for our use case is that words are not strictly aligned on a line and thus reading line-by-line is not possible. Compared to other words clouds, such as Wordle [VWF09] the words are still easily readable, because all words are aligned horizontally.

5.4.2 User Evaluation

In the user evaluation we wanted to examine whether the text representation form (full-text, key sentences, key phrases) had an influence on the correctness of the labels assigned to the documents and the time required for labeling. Moreover we wanted to examine the influence of the potential mislabelings on different classifiers. In particular we tested the following hypotheses:

- H1** The time required for labeling key phrases or key sentences is significantly less than for labeling full-text documents.
- H2** There is no difference in the number of correct labels between key phrases, key sentences and full-text.
- H3** There is no difference in classifier accuracy when using labels generated in the key phrases, key sentences or full-text condition.

Furthermore, we were interested in the users' perception of their performance when using the different types of representation. Also we wanted to find out whether they preferred a particular representation form or disliked another one. More specifically, beyond the hypotheses enumerated above we investigated the following assumptions:

- A1** Users prefer the key sentence representation to the other two, because it is more familiar than the word clouds and more concise than the full-text representation.
- A2** Users feel performing most accurately using the full-text representation, because it contains most information.

5 Experiments

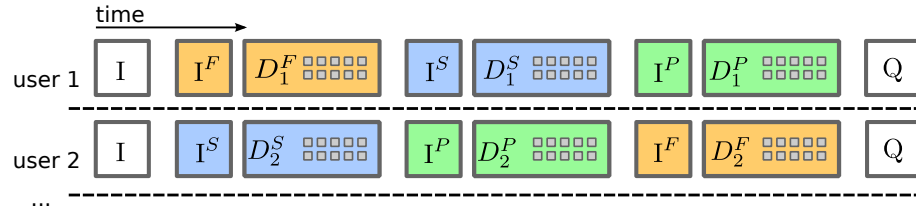


Figure 5.14: Overview of the evaluation procedure, **I** abbreviates an Introduction step, **F** (full-text), **S** (key sentences), and **P** (key phrases) denote the different conditions. **Q** abbreviates the final questionnaire.

Design

We used a within-subjects design, i.e., each user performed all conditions. The independent variable is the text representation form with three different levels (full-text **F**, key sentences **S** and key phrases **P**). We measured task completion time and correctness of the task (dependent variables). The task completion time is measured as the time difference between the user first seeing the document and finishing the assignment for this document. Correctness of the task is calculated as the number of correct user labels by comparing the user labels to the ground truth of the annotated corpus.

Procedure

Figure 5.14 gives an overview of the evaluation procedure. For each participant, the study started with an introduction of the task and with an example document for each condition. Then the participant had time to ask questions. Thereafter the participant was asked to fill out a demographic questionnaire. Then, the three trials on the computer started. The sequence of conditions (**F**, **S** and **P**) and the documents were randomly chosen from the data set (see section 5.4.2 for details).

For one trial (10 subsequent documents) the presentation form was the same (e.g., all documents presented as full-text). Each trial started with an introductory screen. After the participant had clicked the 'OK' button, the measurements started. We measured the task completion time (the time between the two subsequent clicks on the 'OK' button) and collected the labels that the participants assigned to the presented articles. For each of the three conditions, we computed the mean value for the completion time and counted the number of correct labels. Thus, for each participant i , $1 \leq i \leq 37$ we obtained one single value for the number of correct labels l_i^c , and completion time t_i^c per condition $c \in \{\mathbf{F}, \mathbf{S}, \mathbf{P}\}$.

The study finished with a questionnaire for the participants in which we asked questions about the perceived overall task difficulty, and the perceived stress. Further, for each presentation form, we asked the participants to rate the perceived helpfulness, speed and difficulty and how much they liked to work with the representation. The users

5.4 Experiment 3: Document Representation for Efficient Labeling

rated these aspects on a 5 point Likert scale. In the following the questions are listed (translated from German) and the meaning of the different values is noted in brackets.

- 'How helpful was *this presentation form* for finding the correct topic?' (1-not helpful, ..., 5-very helpful). We abbreviate this condition with **helpfulness**.
- 'How difficult was it to find the correct topic with *this presentation form*?' (1-difficult, ..., 5-very easy). We abbreviate this condition with **difficulty**.
- 'How fast could you identify the correct topic with *this presentation form*?' (1-slow, ..., 5-fast). We abbreviate this condition with **speed**.
- 'Was it stressful to identify the topics for the texts using *this presentation form*?' (1-very stressful, ..., 5-not stressful). We abbreviate this condition with **stress**.
- 'How much did you like to use *this presentation form*?' (1-did not like it, ..., 5-liked it very much). We abbreviate this condition with **like**.
- 'Were there enough hints for the topics in *this presentation form*?' (1-too little, ..., 5-enough). We abbreviate this condition with **hints**.
- 'How difficult was it, to map the texts to the topics?' (1-very difficult, ..., 5-very easy). We abbreviate this condition with **overall difficulty**.
- 'How stressful was the task in general?' (1-very stressful, ..., 5-not stressful). We abbreviate this condition with **overall stress**.

Further we asked the participants two open questions: (i) Which text representation forms could you imagine to make texts accessible in a fast way? (ii) How could we improve the presented text representation forms in your opinion? Here the users were no given any choices to select by could write their answer in free-form text.

Test Material

We used a German news corpus from the Austrian Press Agency consisting of 27570 news articles from the year 2008 (see section 5.1.4 for description of the data set). We chose to use 3 year old news articles to reduce the effect of remembering recent news. The corpus is fully labeled, i.e., each news article is annotated with one of the five classes "economy", "sports", "culture", "politics", "science". The articles are nearly equally distributed over the classes. The length of the articles varies between 2 and 2720 words, the average length is 247.2 words.

We investigate how often the class names (e.g. politics) occur in the extracted key words. For 616 of 27570 ($\approx 2\%$) of the documents an extracted key phrase is equal to a word describing the class. This means, it is very unlikely that users can use the class name as a clue for labeling.

5 Experiments

We chose the longest articles of the corpus for our experiment, i.e. the articles longer than the third quantile (> 337 words) without the statistical outliers (articles with > 655 words), see figure 5.15. This leaves 6328 articles for the experiment, 1508 in class "culture", 1023 in "economy", 1409 in "politics", 1457 in "science" and 931 in "sports".

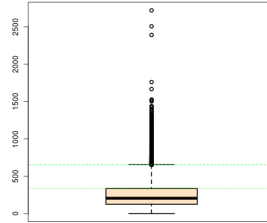


Figure 5.15: Box plot of the word distribution in the data set. Green (horizontal) lines bound the region of the documents used for the experiment.

For each condition a set of documents is presented to the user, we chose to take 10 documents per condition. The document set for a condition is denoted as $D^{\mathbf{F}}$, $D^{\mathbf{S}}$, $D^{\mathbf{P}}$ respectively. For a user k the sets are denoted as $D_k^{\mathbf{F}}$, $D_k^{\mathbf{S}}$, $D_k^{\mathbf{P}}$. All articles in all document sets are distinct, i.e., no user gets one document twice. For articles in set $D^{\mathbf{S}}$ key sentences, for articles in set $D^{\mathbf{P}}$ key phrases and named entities were extracted as described in section 5.4.1. The key sentences and the full-text were displayed in a normal text windows (see figure 5.16 for an full-text example and figure 5.17 for key sentences). The key phrases and named entities were laid out with the tag layout algorithm described in section 5.4.1. In order to visually distinguish key phrases and named entities, the key phrases were colored black and the named entities were colored blue. An example for a key phrases representation is shown in figure 5.18.

Participants

37 German-speaking volunteers participated in the evaluation, 18 females and 19 males. 23 of the participants were technical professionals while 14 were experts of other domains. The age of the participants ranged from 25 to 58 years (average 32.5 years).

Environment

The participants were tested in a calm environment without noise distractions or additional attendees. The task was performed on a Dell Latitude e650 notebook running Windows XP Professional. The notebook was equipped with an Intel Core Duo 2.26 GHz and 3 Gb RAM. The display resolution was 1440 x 900 pixels. All users were required to use the USB mouse (and not the touch pad).

5.4 Experiment 3: Document Representation for Efficient Labeling

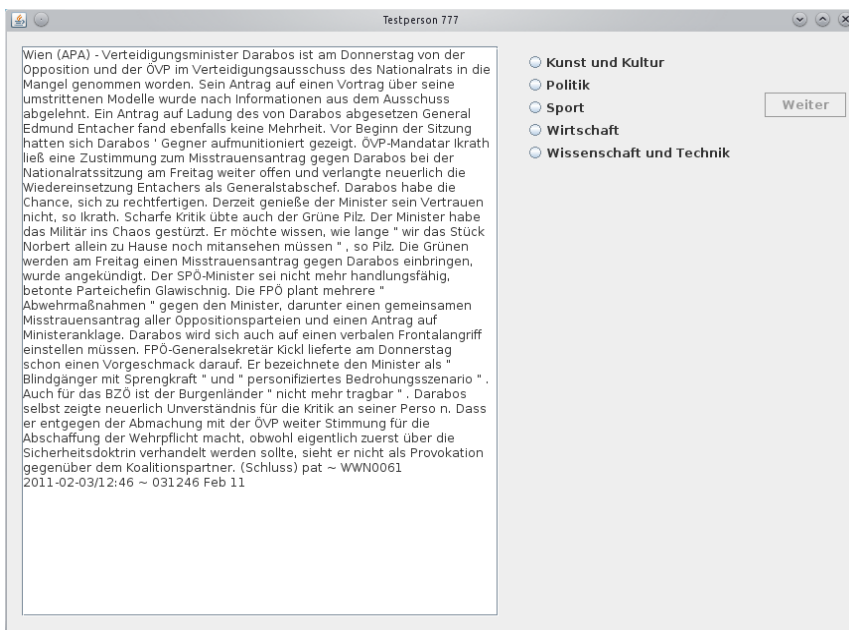


Figure 5.16: Screenshot of the application for the full-text condition **F**. Data is extracted from the German test corpus.

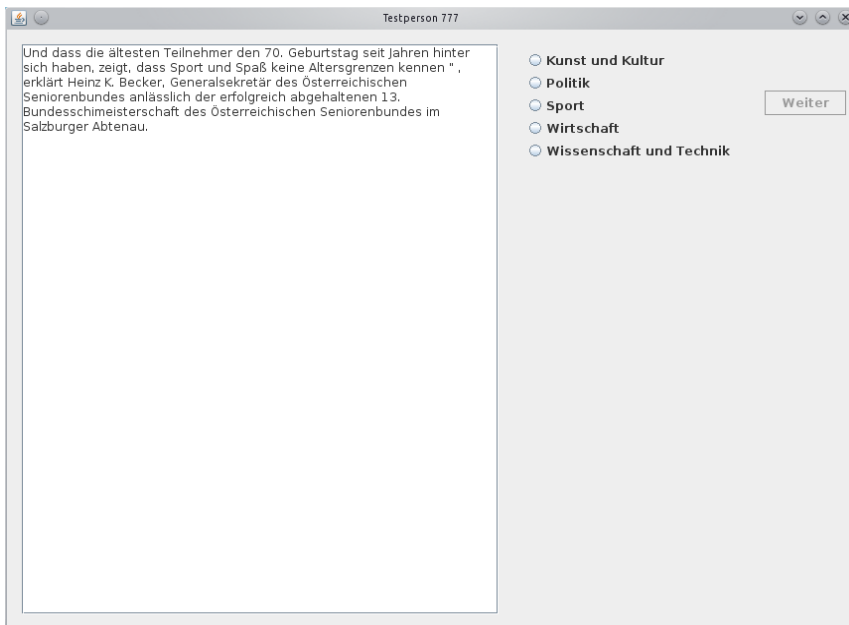


Figure 5.17: Screenshot of the application for the key sentences condition **S**. Data is extracted from the German test corpus.

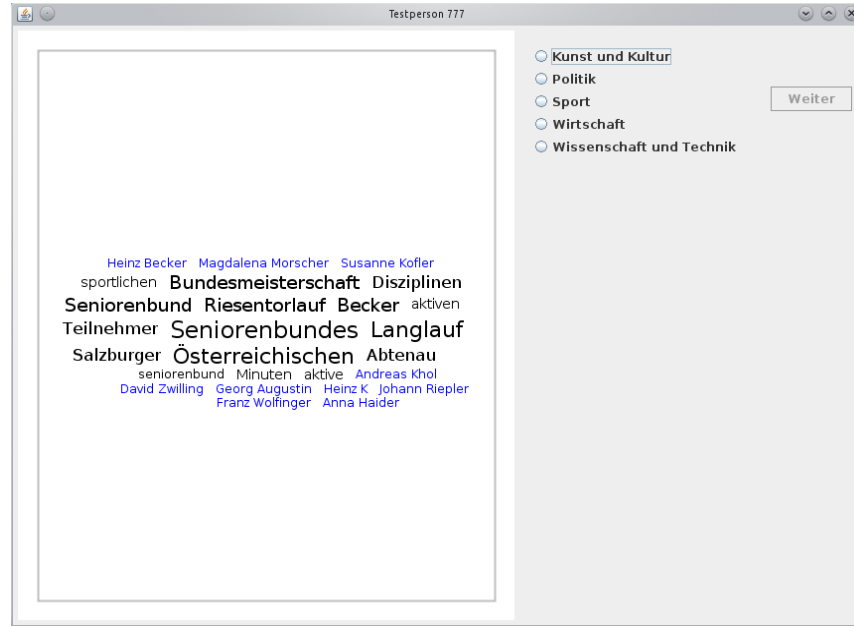


Figure 5.18: Screenshot of the application for the key phrases condition **F**. Data is extracted from the German test corpus. Named entities are colored blue.

5.4.3 Results

In this section we present (i) how we tested the three hypotheses enumerated at the beginning of section 5.4.2 (measured performance), (ii) the results of the quantitative evaluation of the questionnaire (perceived performance), (iii) answers to the questionnaire’s open part (suggestions and improvements).

Measured Performance

Table 5.18 and figure 5.19 summarize the measures for the number of correctly labeled examples and the task completion time. Altogether, the users assigned 290 correct labels in the full-text condition, 281 in the key sentences condition and 305 in the key phrases condition. In total 370 documents (10 documents per user, 37 users) were labeled in each condition. In the following sections we describe in detail how we tested the three hypotheses enumerated at the beginning of section 5.4.2.

Influence on Labeling Accuracy We tested whether the difference in the correct number of labels reported in table 5.18 are significant (Hypothesis H1). The correct number of labels is denoted as l_i^c for person i and condition c . As can be seen from the histograms of figure 5.20 the three variables l^f , l^s and l^p seem to be not normally distributed and thus the precondition for performing ANOVA or paired T-tests is not satisfied. However,

5.4 Experiment 3: Document Representation for Efficient Labeling

Table 5.18: Overview of labeling time and number of correct labels (out of 10) for each condition. Values averaged over all users, showing mean and standard deviation.

	full-text	key sentences	key phrases
correct labels	7.84 ± 1.24	7.59 ± 1.38	8.24 ± 1.23
completion time [s]	19.9 ± 13.8	10.7 ± 4.4	10.4 ± 4.1

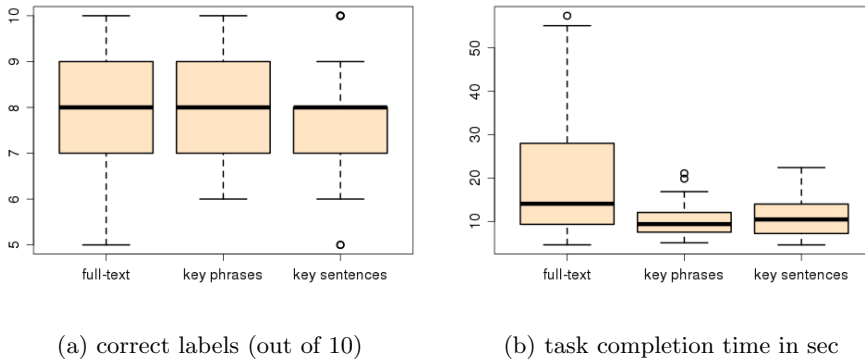


Figure 5.19: Box plots for task completion time and number of correct labels averaged over all users

we still tested the variables l^f , l^s and l^p for normal distribution using the Shapiro-Wilks test. All variables are not normally distributed, assuming $\alpha < .05$. Therefore, we tested on equal means with Wilcoxon rank sum test for unpaired samples. The null hypothesis for the test was that the means are equal, we set $\alpha = .05$. No difference in the mean values was found between full-text and key phrases ($W = 563, p = .177$) and between full-text and key sentences ($W = 754, p = .441$). Comparing key sentences to key phrases we found a significant difference in the mean values ($W = 504, p = .46$).

Summing up, we found that users assigned significantly less correct labels when using the key sentence representation of the documents, but performed equally well with the full-text representation and the key phrases.

Influence on Labeling Time We tested further whether the differences in task completion time reported in table 5.18 are significant (Hypothesis H2). The average time for labeling is denoted as t_i^c for person i and condition c . As can be seen from the histograms of figure 5.21 the three variables t^f , t^s and t^p seem to be not normally distributed and thus the precondition for performing ANOVA or paired T-tests is not satisfied. However, we still tested the variables t^f , t^s and t^p for normal distribution using the Shapiro-Wilks test. All variables are not normally distributed, assuming $\alpha < .05$. Therefore, we tested

5 Experiments

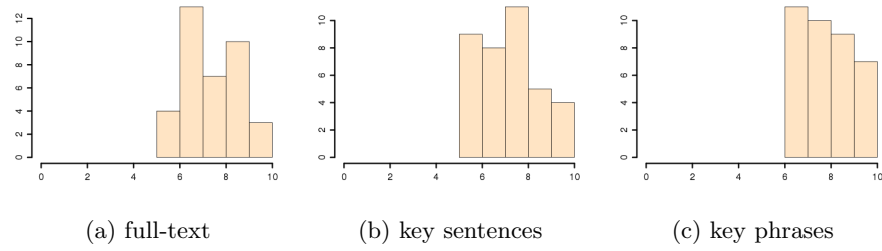


Figure 5.20: Histograms of the number of correct labels averaged over all users

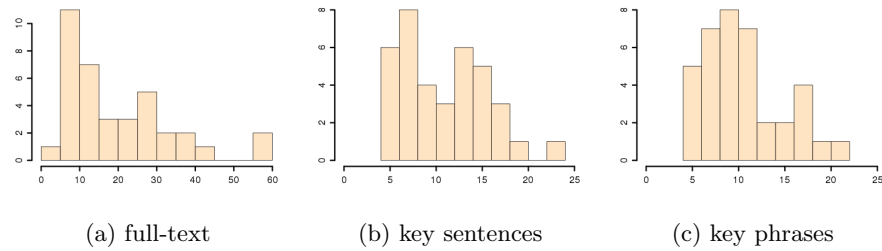


Figure 5.21: Histograms for the task completion times averaged over all users

on equal means with Wilcoxon rank sum test for unpaired samples. The null hypothesis for the test was that the means are equal, we set $\alpha = .05$. No difference in the mean values was found between the full-text and key sentences ($W = 705$, $p = .830$). On the contrary, we found a significant difference comparing full-text and key phrases ($W = 956$, $p = .003$) and full-text and key sentences ($W = 982$, $p = .001$).

Summing up, we found that users labeled the items significantly faster when using the key sentence or the key phrases representation than when using the full-text representation of the documents.

Influence on Classifier Accuracy As reported in section 5.4.3 we found that users labeled less accurately when using the key sentence representation of the text documents. We further wanted to test, whether this mislabeling would have an influence on classifiers trained on the erroneous labels (Hypothesis H3). To do so, we created two different training data sets for each condition, resulting in six different training data sets. Both training sets for one condition contained the documents processed by all users in this condition, one was extended by the original labels (the ground truth) and the other one was extended by the user labels. We further created an evaluation data set of 6000 randomly selected items from the data set. None of the evaluation items was contained in any of the training data sets. We trained various classifiers on both training data sets for each condition, and evaluated the trained classifiers on the evaluation data set. a_o^c

5.4 Experiment 3: Document Representation for Efficient Labeling

denotes the accuracy of the classifier trained on original labels, a_u^c denotes the accuracy of the classifier trained on user labels for condition c . We used the following classifiers:

- Bagging with Decision Stumps (denoted Bagging-DT) from the Mallet machine learning library [McC02]
- AdaBoost with Decision Stumps (denoted Adaboost-DT) from the Mallet machine learning library [McC02]
- Naive Bayes from the WEKA machine learning library [HFH⁺09]
- Hyperpipes from the WEKA machine learning library [HFH⁺09]
- Linear Support Vector Machines from the LibLinear library [FCH⁺08] (described in section 2.2.5)
- K-Nearest Neighbor classifier (denoted KNN-10 for $k=10$, and KNN-20 for $k = 20$, described in section 2.2.3) (own implementation)
- Class-feature-centroid classifier [GZG09] (denoted CFC, described in section 2.2.6) (own implementation)

Table 5.19: Comparing classifier accuracy when trained on original labels (a_o) versus trained on user labels (a_u)

classifier	full-text		key sentences		key phrases	
	a_o^f	a_u^f	a_o^s	a_o^s	a_o^p	a_u^p
KNN-10	0.76	0.72	0.77	0.73	0.76	0.73
Bagging-DT	0.45	0.45	0.51	0.48	0.47	0.45
LibLin	0.80	0.74	0.80	0.76	0.79	0.74
KNN-20	0.75	0.71	0.76	0.73	0.76	0.72
Adaboost-DT	0.36	0.41	0.39	0.38	0.33	0.31
NaiveBayes	0.81	0.77	0.78	0.76	0.79	0.76
CFC, b=2.3	0.78	0.73	0.78	0.73	0.78	0.72
Hyperpipes	0.78	0.72	0.77	0.71	0.77	0.67

Table 5.19 reports the accuracy of the classifiers on the evaluation data set. Not surprisingly, the accuracy of the classifier trained on user labels was lower in nearly every case than when trained on the original (ground truth) labels. This is because the ground truth was labeled by domain experts and we did not explicitly communicate the rules for assigning an article to a specific category. Thus, for the boundary articles, i.e., news about a politician attending a sports event, the decision whether the article belongs to category "sports" or "politics" was subjective. Because all articles were randomly selected and aligned to the three conditions this effect is likely to occur equally often in all conditions. The one exception is the Adaboost classifier in the full-text condition. However, this is also the classifier that performs worst for this classification task.

5 Experiments

Table 5.20: Comparing original labels and user labels: Difference in number of correct labels and classifier accuracy (mean and standard deviation)

	full-text	key sentences	key phrases
Δ correct labels	71	80	65
Δa	0.034 ± 0.037	0.034 ± 0.017	0.040 ± 0.022

Table 5.20 reports the differences in classifier accuracy averaged over all classifiers for the three conditions. When using the user-labels the accuracy decreases by less than 4% in all conditions. The difference in accuracy for the key phrases seems to be larger ($\Delta a^p = 0.040$) than for the sentence and full-text conditions ($\Delta a^s = 0.034$, $\Delta a^f = 0.034$). We investigated whether these differences are statistically significant. First we tested the variables Δa^f , Δa^s and Δa^p for normal distribution using the Shapiro-Wilks test ($\alpha = 0.05$). The two variables Δa^s and Δa^p follow a normal distribution, but Δa^f does not. This means, the preconditions for calculating ANOVA or paired t-Tests was not fulfilled. Therefore we used the Wilcoxon rank sum test for unpaired samples to compare the mean values using $\alpha = .05$. We found no significant difference between any of the conditions, the test statistics are as follows: full-text vs.key phrases $W = 39$, $p = .462$, full-text vs.key sentences $W = 34$, $p = .833$, key sentences vs.key phrases $W = 29$, $p = .753$.

To sum up, we found no influence of the different representation forms on classifier accuracy.

Perceived Performance

In this section the qualitative and quantitative results of the questionnaire analysis are presented. Table 5.21 gives an overview of averaged values of the perceived helpfulness, speed, difficulty, stress, hints, and how much the user liked the respective text presentation form. A detailed description of the questions and the scale can be found in section 5.4.2. Note, that the second column shows the values for the overall task, which the participants had to rate first. This means after the experiments they were first asked how difficult and stressful they perceived the experiment as a whole. From table 5.21 we can draw the following conclusions:

- The *full-text* condition was perceived as most helpful and least difficult. Further, participants felt that it presented nearly all information they needed to complete the task (hints). Also full-text was liked the most.
- Participants perceived themselves as performing fastest in the *key phrases* condition, however they were more stressed and liked it less than the other two conditions. Further, the *key phrases* condition gave them the least hints on the topic of the document.

Table 5.21: Overview of questionnaire answers. Results are averaged over all participants. Showing mean and standard deviation. Best values (least stressful, fastest, ..) for each row are marked bold. * one missing value in the data set was replaced by the median

question	overall task	full-text	key sentences	key phrases
helpfulness	–	4.1 ± 1.0	3.9 ± 1.0	3.4 ± 1.1
speed	–	3.6 ± 1.0*	3.9 ± 1.0	4.0 ± 1.0
difficulty	3.7 ± 0.6	4.1 ± 1.0	3.8 ± 1.2	3.4 ± 1.2
stress	4.1 ± 0.8	3.4 ± 1.3	3.9 ± 1.0	3.4 ± 1.3
hints		4.7 ± 0.8	3.8 ± 1.1	3.2 ± 1.1
like	–	3.7 ± 1.2	3.5 ± 1.4	3.4 ± 1.4

- The *key sentence* representation caused least stress for the participants. Further, participants felt that they were nearly as fast in this condition as in the key phrases (fastest) condition.

To sum up, we can conclude that users preferred full-text, were least stressed by the key sentences and found that they performed fastest in the key phrases condition.

Suggested Improvements

We asked two open questions dealing with suggestions to improve the representation forms or find substitutes as well as possible amendments. Not all participants answered both or either question. The question "How could the representation be improved?" was answered by 57% of all participants while 65% of them had suggestions on "Which other representation could be used to allow a quick understanding of information in texts?" As most of the answers headed into similar directions we defined three categories for all given answers to both open questions. The categories for the first question are:

- Formatting and Highlighting
- Content
- Structure

The first category includes formatting issues like paragraphs, font, or the font size. The second category covers the represented content as for example selected words or sentences. The third category is about the structure of the given representation form itself; e.g. the order of the given information. The categories for the second question are:

- Formatting and Highlighting
- Representation
- Visualizations and Graphics

5 Experiments

The first category again deals with formatting issues. The second category includes representation structures like tree maps and the third category covers visual aids like tables or pictures.

For the first question we found that 14 answers contained suggestions about formatting issues, three dealt with the represented content, and four suggested different structuring of the text examples. Examples for answers to the first category of the first question are: "formatting of the full text examples, paragraphs, font", "bigger font, different font". The second category of the first question contained suggestions like: "Key sentences: they are hard to read. Relations between the sentences are not logical". The third category of the same question had answers including: "Group words by the kind of words at key phrases (named entities, names, adjectives)".

The second question contained eight answers to the first category. 12 answers were related to the first category Formatting and Highlighting and four answers suggested an alternative visual representation of the text information. Examples for the first category are: "Highlight keywords in long texts", "Formatting (breaks, fat, italic, colours!)". The second category contains: "A combination between full-text and key phrases (more key phrases)" and "Hypertree, TreeMap". The third category of the second question includes: "Pictures, Graphics, Icons", "No idea (pictures, tables, graphics)".

5.4.4 Discussion

In this section we discuss our hypotheses outlined at the beginning of section 5.4.2 in the light of the results of the previous section. The evaluation showed that users can label key phrases twice as fast but with the same accuracy as full-text documents. Labeling of key sentences is fast too, but the labeling accuracy is significantly lower than in the full-text condition. This means we can accept hypotheses **H1**: that a compressed representation leads to faster decisions, regardless whether this decision is correct or not.

Hypothesis **H2** must be rejected, there is a difference in the number of correct labels when varying the representation form. More specifically, users are most accurate when using full-text or key phrases, indicating that the TextRank algorithm for keyword extraction performs well in filtering out information irrelevant for text categorization while keeping the information required to identify the category. On the contrary, the labeling accuracy for key sentences is significantly lower, indicating that key sentences are less informative on average, obviously either irrelevant or ambiguous sentences are extracted.

In our experiments we found no influence of this different labeling accuracy on classifier performance, thus confirming hypothesis **H3**. This might be due to the noise tolerance of the used classifiers and the practically low amount of noise. In our experiment, it makes no difference for the classifier whether 65 or 80 out of 370 documents are labeled incorrectly. We expect this difference to become significant when the number of training items (and thus the number of mislabeled items) increases.

5.4 Experiment 3: Document Representation for Efficient Labeling

Analyzing the questionnaire lead to interesting results. We found that on average users preferred the full-text to the key sentences and key phrases. Furthermore, they found full-text and key phrases more stressful than key sentences. We suppose that this is because, (i) they are used to read sequences of sentences (as opposed to word clouds) and (ii) the texts are shorter than full-text, i.e. less text to read in total. Further, users felt labeling fastest in the key phrases condition and slowest in the full-text condition. Thus, our assumption **A1** could only be partly confirmed.

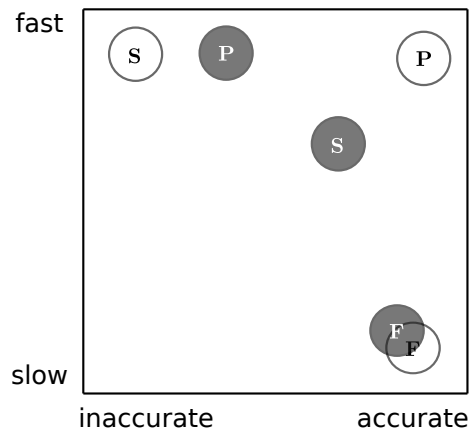


Figure 5.22: Infographics showing perceived (filled circles) and measured performance (empty circles) for all conditions (**P** – key phrases, **S** – key sentences, **F** – full-text)

Interestingly, in some aspects the perceived performance does NOT conform to the measured performance. First, users felt that they were least hints in the key phrases condition which was also perceived least helpful, but our measurements show, that they performed as accurately as in the full-text condition. Thus, our assumption **A2** can be confirmed. Second, users felt to have more hints in the key sentences condition and found the presentation more helpful than the key phrases condition, but according to the measurements performed worst in the key sentence condition. Note, that participants on average did like the key phrases condition less than the others. We assume that this is because the former was unfamiliar. For applications this would mean that users need to be convinced of the helpfulness of the key phrases representation and get used to it. The infographics in figure 5.22 summarizes the differences between perceived and measured values. (Note that this infographics visualizes tendencies, not accurate values).

The answers to the open questions of the questionnaire gave interesting hints on how to improve representation of text and led to new ideas on how to structure information in texts. The utmost answers to question one, how to improve the given representations, regarded formatting and highlighting. We therefore believe that a way to change font and size of texts – like buttons to control the font size – could be of much help to faster understand the text information. Furthermore, coloring keywords could help users to find relevant information faster. To satisfy suggestions regarding the content other ways

5 Experiments

to select words and phrases could be tested. As for the structuring of the representation forms most of the answers suggested a different ordering of key phrases. Ordering of the words could be made selectable like nouns and names first. The second open question of the questionnaire showed that many users had similar ideas about highlighting important word in full texts. This could be met by giving users the possibility to automatically highlight all nouns or names. Users also had suggestions on how to combine different representation forms. Combinations of key phrases and full-text, like full text with selected key phrases – on demand could be useful. Also, interesting ideas emerged regarding visualizations and graphics. Tables containing words of a category or a graph showing the count of keywords could be shown.

Summing up, our evaluation shows that: *Key phrases are a fast and accurate representation for document labeling. In fact, users labeled key phrases twice as fast and as accurately as full-text documents. Further, the questionnaire shows, that users did not trust their labellings with the word cloud representation, although they performed most accurately.*

5.4.5 Summary

We investigated two different condensed representations of text, key phrases and key sentences, for the purpose of faster document labeling. Both representation forms can be generated in a fully automatic way. In a user evaluation we compared the labeling accuracy and time of the users when using these condensed representations to the baseline, the full-text representation of the texts. Our evaluation shows that the users labeled key phrases twice as fast but as accurately as full-text documents. This finding points toward a feasible way to decrease the time and cost for the generation of training data. Key phrases represented as word clouds for labeling can be easily combined with other approaches such as active learning.

In one sentence the outcome of this experiment can be summarized as follows:

Key phrases, automatically extracted by the TextRank algorithm, are a fast and accurate representation for document labeling.

5.5 Experiment 4: Visual Hypothesis Generation

In this section a prototypical implementation of a Visual Analytics tool for generating classifier hypotheses from scratch is described. All previous experiments considered the case when the set of classes is already known beforehand. In this experiment the case is explored where the classes are not known and will only emerge when analyzing the data at hand.

The question that will be answered by the experiment in this section is

Can pure data visualizations be used to allow domain experts to generate their own classifiers?

In order to answer this question the following steps are taken:

- The prototype is built using existing visualizations and data processing modules.
- The application is applied on a standard text set and results are recorded.

The experiment presented in this section has been published in [SSG10].

The proposed application supports the following user-centered approach to text classification: (i) structure the data fully unsupervised, (ii) present the data to the user using information visualization techniques, (iii) support the user in exploring the data and the precomputed structure, (iv) support the user in employing an additional structure (supervised) on the data, (v) visualize the imposed additional structure and allow the user to assess the quality and the appropriateness for the task at hand.

For the automatic structuring of the data clustering techniques are employed. Classification techniques are used to impose an additional user-generated structure. In this application, the data is automatically analyzed, structured and then presented in an interactive visualization. The interactive visualization can then be used to build a classifier from scratch, i.e. defining the classes as well as the training items for each class. Noteworthy, the user does not need to know the classes of interest beforehand, but they can be created, changed and deleted as demanded. Furthermore, this combination of interactive visualization of the data and a classifier further allows a personalization of the classifier. Depending on the data and what a specific user is interested in the data, the classes, the training data and finally the classifier can be built. From the data mining perspective, the application implements a user-centered approach to on-line, multi-label, multi-class text classification.

5.5.1 Procedure

In the proposed application the analytic task of the user is supported by fully-automatic processes. This section describes the overall process and necessary visualization modules for the application.

Process Overview

Figure 5.23 depicts the process of the proposed application. Analytic processes performed by the user (gray round boxes) alternate with fully-automatic processes (white boxes). In the following the necessary building blocks (white boxes) are described in detail.

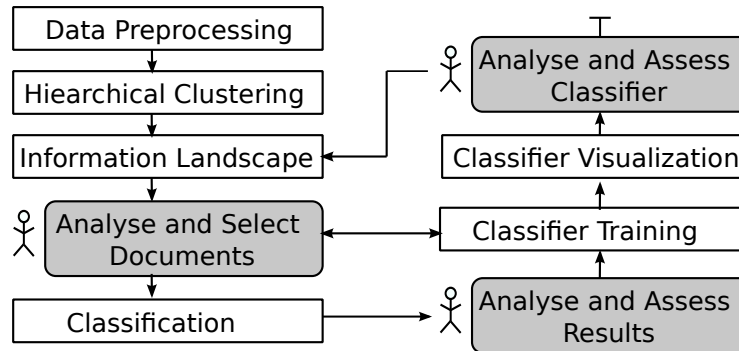


Figure 5.23: Process overview of the combined approach: Analytic processes performed by users (grey round boxes) are supported by fully-automatic processes (white boxes). The (re-)training of the classifier is done iteratively and stops when the user is satisfied with the quality of the classification hypothesis.

Note that the majority of the applied algorithms and visualizations are implemented within the KnowMiner knowledge discovery framework [KSM⁺09] and VisTools visualization library [SKM⁺09].

Data Preprocessing The data is preprocessed and vectorized with an information extraction and vectorization module based on OpenNLP³. The POS-tags are used to construct the noun vector space. The result of the data preprocessing steps are documents represented as feature vectors. For details of the preprocessing step see section 2.3

Hierarchical Clustering A hierarchical clustering algorithm is applied on the feature vectors of the documents to reveal the inherent structure of the document space. An adapted K-Means algorithm [Mac67] is recursively applied using the cosine similarity to measure similarity between documents. Cosine similarity is known to perform well for text data [ZK02]. The recursive application of the K-Means clustering algorithm creates a cluster tree, in which the leaves represent single documents. The resulting cluster hierarchy is suitable for browsing the document set and can be thought of as a virtual table of contents.

³<http://opennlp.sourceforge.net>

Creating the Information Landscape The information landscape visualization is computed from the topical cluster tree by a combination of **Force-Directed Placement (FDP)** algorithms and spatial tessellations. First, the top-level cluster centroids (the children of the root cluster) are positioned inside a rectangular area using **FDP** (FDP). This rectangular area corresponds to the overall available display space for the information landscape. The similarity of the cluster centroids used by the FDP algorithm is calculated as the cosine similarity of the document vectors. Second, a Voronoi subdivision is calculated using the centroids as generator points for the Voronoi regions yielding polygonal regions representing a document cluster. This process is recursively applied inside the polygonal region for all sub-clusters of a specific cluster. The leaves of the bottom-most clusters (documents) are placed within the Voronoi area of their parent cluster using the same **FDP** method.

Text Classification We applied an adapted **KNN** algorithm for multi-label text classification. As similarity we use the cosine similarity on the **TF-IDF** weighted vector-space representation of the documents. The output of the classifier for each classified document is a list of classes and a confidence value for each class. The visual analytics application is in principle independent of the specific classifier, as long as multi-label classification is supported.

The Classification Panel The classification panel provides the interface to the classifier. It gives an overview of the available classes, the associated training documents and classification results. Moreover it incorporates the Class Radial Visualization to see the trained classifier at a glance – all classes and associated trainings documents.

Class Radial Visualization in the Multi-Label Case For single-label classification, the case for which the classifier visualization was designed, the location of a test item shows the uncertainty of the classifier for this item. In single-label classification one item can only belong to one class. In multi-label classification it is possible that one item belongs to more than one class. In this case, the Class Radial Visualization shows an overview how many documents are assigned to multiple classes (how much and to which). When the mouse is moved over an item, lines are drawn connecting the item and all classes the item belongs to. The thickest line (red) indicates the connection to the class the item belongs with the highest confidence. The classification assignments for items located at the same position are aggregated, the item is treated as a new hyper-item.

5.5.2 Interaction Methods

The clustering view (information landscape) and the classification panel support different tasks. Table 5.22 gives an overview of possible interactions that can be performed in the different interfaces.

5 Experiments

Table 5.22: Overview of the tasks that can be performed from both interfaces information landscape (IL) and classifier window (CW).

Task	Invoked in		Results in
	IL	CW	
create class	✓		A new class is created from the selected documents. The classifier is retrained.
delete from class	✓	✓	The selected documents are removed from the class. The classifier is retrained.
delete class	✓ ¹	✓	The selected class is deleted. The classifier is retrained.
classify		✓	The classifier predicts the selected documents. The prediction is presented to the user in a table.
inspect documents	✓	✓	An information landscape of the selected documents is displayed.
inspect class		✓	An information landscape of the training documents of the selected class is displayed.
inspect classifier		✓	The classifier visualization for the training data is displayed.

¹ Deleting of classes can not explicitly be invoked from the information landscape, but a class is automatically deleted if all its training documents are removed.

5.5.3 Results

We performed our experiments on the Reuters-21578 text collection (see section 5.1.3). The hierarchical clustering results in 10 top-level clusters as shown in figure 5.24a. This structure of the information space is purely unsupervised. The information landscape in figure 5.24a gives an overview of the entire text collection, showing clusters of similar documents and associated labels. The user can investigate the cluster hierarchy and get an insight in the overall content of the collection.

The user might be interested in other partitions of the data set which are not detected by the unsupervised methods. For example, the user might want to distinguish the categories 'politics', 'computers', 'cars', 'sports' and 'planes'. First, these categories are not explicitly represented, they only exist as a mental model in the user's mind. While investigating the information landscape the user might come across documents that belong to one of these categories. The user can then select these documents (as shown in figure 5.24a) and can create a new category from the selected documents. In the background, the selected documents are added to the classifier as new training data

5.5 Experiment 4: Visual Hypothesis Generation

for the specific category (if the category does not yet exist in the classifier, it will be created). After repeating the steps 'investigation" and 'adding training documents to the classifier" the user might have found example documents for each of the categories of interest.

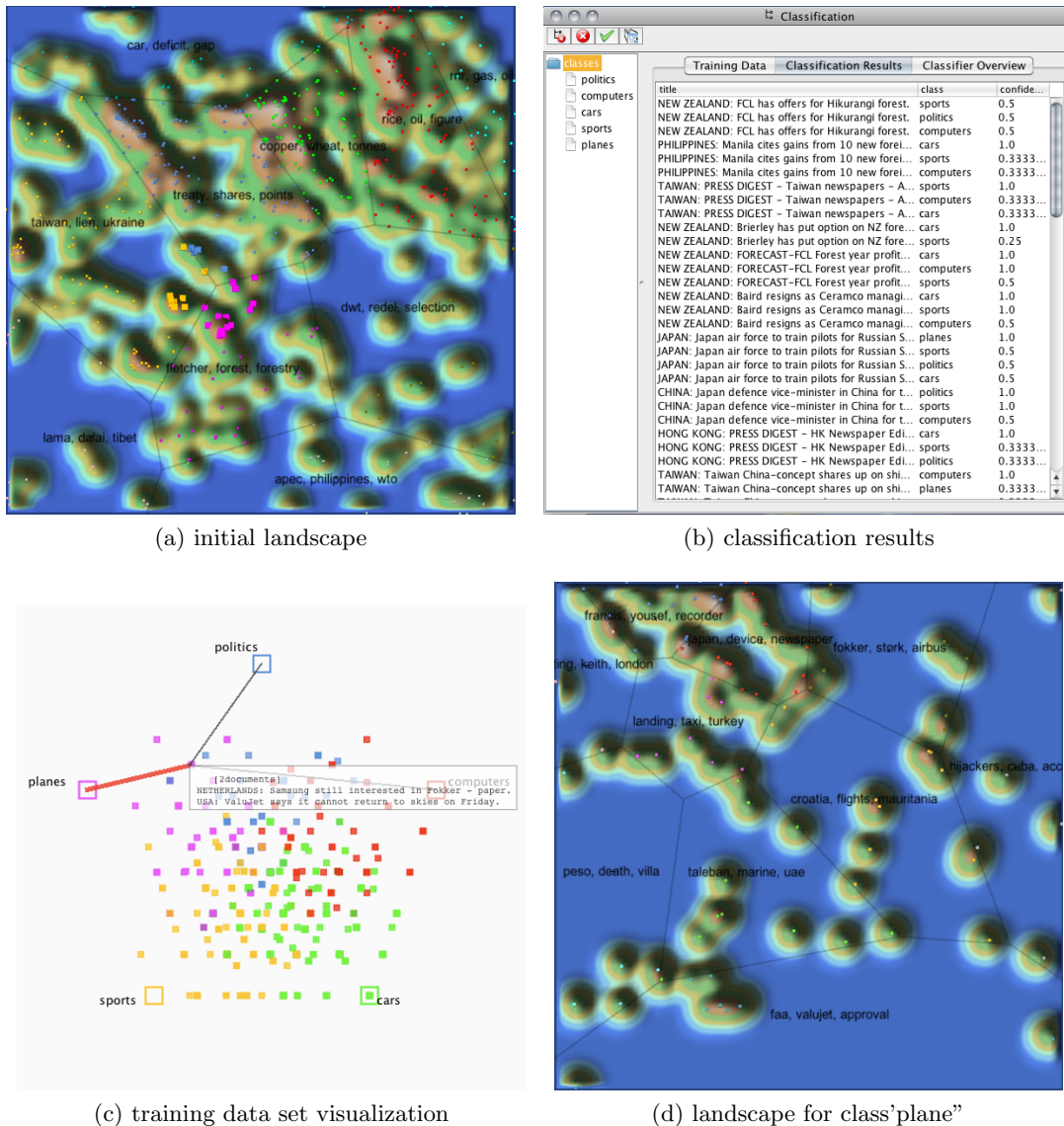


Figure 5.24: Screenshot of the text classification application. (a) information landscape of the data set with documents selected by user, (b) classification results for selected documents (classifier has been trained beforehand) (c) visualization of the training data set, (c) landscape visualization of training data for class 'plane'

5 Experiments

The user might then be interested of the current available documents for each category and the quality of the classifier that she has implicitly generated. This information is provided by the classification window. The training document for each class are presented as a list to the user. If the user detects wrongly assigned documents for a category she can simply remove them from the list and the classifier is retrained on the reduced training data set. For assessing the overall classifier quality the user can switch to the classification visualization view as shown in figure 5.24c.

In the figure, it can be seen that there are many documents belonging to more than one class (the central area). Only for the class 'car' there are documents belonging to no other class. Further, there are some documents belonging to exactly 2 classes, these are the documents lying on the imaginary line between the 'cars' and the 'sports' rectangle as well as on the imaginary line between 'cars' and 'computers'. The user might investigate the content of the interesting documents by moving the mouse over the items and eventually discover misclassified items. If the user discovers that the classification model is not in line with her mental model of the categories, e.g., that the categories should be more distinct (i.e. lesser documents in the center of the visualization), the user could select the conspicuous documents and generate a new information landscape in order to further investigate them. Similarly a new information landscape can be generated for all training documents of one category. The resulting landscape is shown in figure 5.24d. Investigating this new landscape might lead to further insights and actions, for instance finding and deletion wrongly assigned training documents.

After cleaning up the classifier by consolidating the training sets, the user might be interested if there are more documents inside the collection that fit into these categories. Back in the information landscape she then selects documents and gets them classified. The classification result for the documents selected in figure 5.24a is shown in figure 5.24b. Then the classification results can be investigated and, in case the classifier correctly classified the documents, can be added to the trainings data set.

5.5.4 Discussion

Generating classifier hypothesis for large dynamic text data repositories is a challenging and time-consuming task. We described a prototypical application, which combines automatic and visualization-based approaches. The user is presented an interactive visualization of the text collection, the information landscape, which is useful for gaining insights into topical structures present in the data set. The newly discovered information is useful for defining the training set of the classifier. The resulting classifier can be evaluated by the means of a classifier visualization and refined further if necessary. We see clear advantages of our approach in the case when the categories are not pre-defined, but emerge during investigation of the document set. However, we also believe that the information landscape is useful for analysis and improvements of existing training sets, for example when the quality of the classifications is deemed unsatisfactory by the user.

5.5.5 Summary

In this section an application to generating classification models from scratch was described. The application was applied to a standard text data set. The proposed application supports the following user-centered approach to text-classification: (i) structure the data fully-unsupervised, (ii) present the data to the user using information visualization techniques, (iii) support the user in exploring the data and the pre-computed structure, (iv) support the user in employing an additional structure (supervised) on the data, (v) visualize the imposed additional structure and allow the user to assess the quality and the appropriateness for the task at hand.

In one sentence the outcome of this experiment can be summarized as follows:

The information landscape as unsupervised data layout embedded in an interactive application allows domain experts to generate classifiers from scratch.

The application has been deployed and applied in two companies, Mimos⁴ and M2N⁵.

⁴www.mimos.my

⁵www.m2n.at

5.6 Experiment 5: Visualizing Text Classification Models

This experiment considers the understanding part of the research questions of this thesis. We applied the Voronoi Word Cloud visualization (see section 4.4 on the internal representation of the CFC text classifier (see section 2.2.6). Thus, some aspects of the internal model of the CFC classifier are accessible to users by the visualization. In this experiment the CFC classifier is trained on the 20NEWS data set (see 5.1.5) and the trained classification model is visualized as Voronoi Word Cloud.

The question that will be answered by the experiment in this section is

In which way can a model of a specific text classifier, namely the CFC classifier, be visualized and made accessible to users?

The experiment presented in this section has been published in [SKG11].

5.6.1 Procedure

This section describes the detailed procedure of applying the Voronoi Word Cloud visualization to text data sets classified with the CFC classifier. We assume that the text data set has already been preprocessed and is represented in a vector-space-model (for details see section 2.3).

The steps to generate the Voronoi Word Cloud for a trained model of the CFC classifier are as follows:

1. Define the border of the surrounding region. An arbitrary convex polygon can be used. Circles and ellipses can be approximated by polygons.
2. The CFC classifier is trained on the data set (or a subset) obtaining feature vectors f_i for each class i .
3. The class feature vectors f_i are placed into the 2D-space using an FDP algorithm. The cosine similarity between two feature vectors f_i and f_j is taken as the high-dimensional distance for the FDP. The results of this placements are 2-dimensional points p_i .
4. A Voronoi diagram is calculated using the projected class feature vectors p_i as generator points obtaining a polygon P_i for each class.
5. For each class feature vector f_i , the k dimensions with the highest values are selected. For each term represented by this dimension, the feature value is added as weight obtaining k keyword-weight pairs for each class. Additionally, the class name is added to the keywords with a weight value larger than the weights of all keywords.
6. The weighted keywords for each class i are laid out inside the polygon P_i for this class using the tag layout algorithm (see section 4.3).
7. The class feature vectors f_i are placed into the 3D-space, and their projection is used to select the background color for polygon P_i from the RGB color space.

5 Experiments

5.6.2 Results

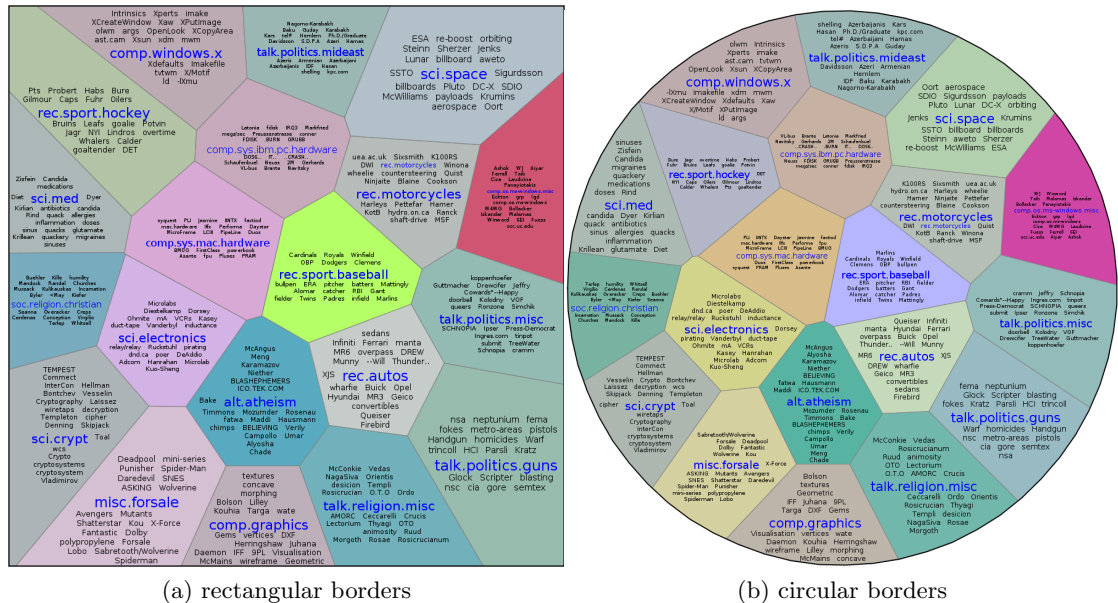


Figure 5.25: Voronoi Word Clouds for the CFC classifier trained on the 20NEWS data set. Different coloring in both examples is caused by non-deterministic version of the FDP algorithm.

The Voronoi Word Cloud visualization of the fully trained CFC classifier on the 20NEWS data set is shown in figure 5.25. Two examples with two different borders (square and circle) are shown. Because the FDP algorithm used in step 3 (projection in 2D) is deterministic, the placement of the class centroids is identical in both cases.

On the contrary, the FDP algorithm applied in step 7 (projection to 3D) is not deterministic, therefore the colors in both visualization differ. The 10 terms with the highest TF-IDF weight in the full data set, i.e. independent of the classes, are 'article', 'people', 'X', 'time', 'way', 'God', 'system', 'anyone', 'something', and 'problem'. These words do not occur in the visualization, i.e. are not important for the classification of the documents. The visualization reveals, that for instance for the newsgroup comp.windows.x, which covers topics of the X-Windows system, the words 'Xdefaults', 'xdm' and 'mdm' are highly informative. Further, one can see that the newsgroups alt.atheism and talk.religion.misc are similar to each other, as they are positioned next to each other and are colored similarly. Obviously their documents cover the same topics, although they do not have one of the 10 most informative words in common. Further, one can judge the quality of the data and preprocessing: i.e. the term 'Cowards"-Happy' in talk.politics.misc is obviously not well tokenized and for the class sci.crypt the terms 'cryptosystem' and 'cryptosystems' are considered as different features.

5.6.3 Discussion

In the light of this thesis this experiments contributes to the understanding part of the research question and can be further extended to allow for adapting a classification model as well. The adaptation mechanisms are not part of the thesis but will be conceptually described in the following:

- The user can remove (irrelevant) words for a class. This will set the according feature in the centroid vector of the CFC model to the value 0 resulting in the classifier further ignoring the word.
- The user can add a word to a class. This will increase the weight of the word either by a user-specific value such that it is ranked among the displayed words. This will result in a changed value in the centroid feature.
- The user can change the weight of a word. This will result in a changed value in the centroid feature.

All these interactions directly manipulate the model of the classifier. For an application it is therefore necessary to also make the classifiers performance accessible to the user, for instance by incorporating the Class Radial Visualization in the application.

Furthermore, one can design an interaction mechanism to allow deeper understanding of the classification model by enabling users to access the relevant underlying data. This interaction mechanism will be described conceptually and is not implemented in the current visualization. Clicking on a keyword could reveal all documents containing the word. These documents can either be filtered by the class for which the word was displayed or all documents can be displayed with a visual indication whether they belong to the chosen class.

5.6.4 Summary

The experiment in this section described a prototypical, classifier-dependent visualization derived from the Voronoi Word Cloud. It allows users to access the internal model of the CFC text classifier. The visualization was applied to the 20NEWS data set. It was shown to be suitable for displaying certain aspects of the classifier's internal model. Further interaction mechanisms were conceptually proposed, which could allow users to also adapt the classifier's internal model.

In one sentence the outcome of this experiment can be summarized as follows:

The Voronoi Word Cloud visualization can be used to asses certain aspects of the internal model of the CFC text classifier.

5.7 Summarizing the Experiments

The findings in the experiments of this chapter can be summarized as follows:

1. The Class Radial Visualization and Confusion Maps can help experts to assess understand arbitrary classification models for text classification tasks.
2. Visual Active Learning with the Class Radial Visualization outperforms classical active learning in all tested conditions and performs never worse than random sampling.
3. Key phrases, automatically extracted by the TextRank algorithm, are a fast and accurate representation for document labeling.
4. The information landscape as unsupervised data layout embedded in an interactive application allows domain experts to generate classifiers from scratch.
5. The Voronoi Word Cloud visualization can be used to asses certain aspects of the internal model of the [CFC](#) text classifier.

“We can only see a short distance ahead, but we can see plenty there that needs to be done.”

(Alan Turing in Computing Machinery and Intelligence)

6 Conclusion and Future Work

This section reflects on the goals set at the beginning of the thesis and assesses the generalizability of the achieved results.

This thesis investigated the usage of interactive visualizations for assessment, understanding, creation and adaptation of classifiers for multi-class, single-label classification tasks. The need for classifier-agnostic visualizations, i.e. visualizations that can be applied to any classifier, was argued. The independence of the visualization from specific classifiers is desired in order to require the user to learn only one visualization and in order to compare classifiers by the means of the visualization.

Thus, common properties of classifiers were identified and two classifier-agnostic visualizations – Class Radial Visualization and Confusion Maps – were designed and implemented. These visualizations were then evaluated in various experiments for their suitability for assessment, understanding, creation and adaptation of classifiers. Various aspects of the classifiers on the class level (i.e. conflicts between classes) and the item level (i.e. confidence of decision) were revealed by the visualizations.

Furthermore, the concept of Visual Active Learning was developed. Visual Active Learning aims – like classical active learning – at minimizing the amount of training data required for achieving a specific accuracy of a classifier. Experiments using the developed Class Radial Visualization showed that Visual Active Learning, i.e. giving the user more power in the active learning process, outperforms classifier-agnostic active learning strategies.

However, classifier-agnostic visualizations can not be applied to visualize the feature level, i.e. which features had which kind of influence for the final decision. This limitation arises from the fact, that knowledge about the used features can not be derived from all classifiers in an explicit form. Therefore, to visualize aspects of the feature level we chose a specific application scenario and classifier. This led to the design and implementation of a visualization for text classification models, called the Voronoi Word Cloud. In order to achieve the desired properties of the layout (space-filling and similarity based), a special layout algorithm for words in arbitrary complex shapes was developed (tag layout algorithm).

6 Conclusion and Future Work

The problem of fast training data generation for classifier creation was addressed in the setting of text classification. Training data has to be generated manually, and the time required for this task heavily depends on the time human labelers need for comprehending the text. We investigated the benefit of text representations alternative to the simple full-text representation. Two text representations were identified, which can be generated fully automatically – key sentences and key phrases. In a user study, these two alternative representations were compared to the full-text representation. The study showed that human labelers performed best in terms of accuracy and time when using the key phrases representation.

In the following (section 6.1) the results of this thesis are discussed in more detail and assessed for their generalizability. In section 6.2 directions to future work are given.

6.1 Assessment

This thesis answers the research question defined in section 1.1: Interactive visualization can improve construction, understanding, assessment, and adaptation of classifiers. In the following, the results of the experiments are critically discussed in detail.

Interactive Classifier-Agnostic Visualization: The identification of principles and limitations for classifier-agnostic visualizations support the design of such visualizations. The two implemented example visualizations show the applicability of these principles. However, as no usability study has been done, the implementation might not be optimal in terms of ease-of-usage.

The performed experiments show that these two visualizations can help experts to assess and understand arbitrary classification models for text classification tasks. However, the visualizations do not support the feature level, because this can not be addressed in a classifier-agnostic way. Thus, the knowledge that can be derived when using these visualizations is limited. Further, the known limitations of visualizations apply: (i) Because color-coding of classes is used in the visualization, the number of classes that can be distinguished by the human eye is limited [Hea96]. (ii) Due to limitations of the display space and the resolution of displays the number of visuals that can be displayed without cluttering effects is limited. The results suggests that, *visualizations can help experts to assess and understand classification models in various ways, but the expressiveness of classifier-agnostic visualization is limited.*

Tag Layout in Arbitrary Convex Shapes: As prerequisite for the Voronoi Word Cloud, a layout algorithm was developed. This allows the space-filling layout of tags or words inside arbitrary convex shapes. Because this algorithm is a heuristic approach to a NP-hard problem, it is not guaranteed to find optimal solutions. Although not being necessarily optimal, certain properties of the layout are guaranteed, e.g. that the most

important tags are laid out first. The results suggests, that *the tag layout algorithm is able to layout words in arbitrary convex shapes, but is not guaranteed to be optimal for all parameter and data combinations.*

Classifier-Dependent Visualization for Text Classification: The Voronoi Word Cloud visualization represents an example of a visualization that addresses the feature-level for the task of text classification. Combined with a specific text classifier, the visualization shows, which features contribute to the classification model in which way and how the trained classes relate to each other in terms of the features used by the classifier. For text classification, the feature-space has usually thousands of dimensions. Thus, it is not feasible to show the relevance of all features, but only the most important ones. Further, the similarity layout of the classes only approximates the true similarity in the high-dimensional space. Thus, insights derived from this layout can also only be approximations. The results suggests, that *the Voronoi Word Cloud visualization can be used to asses certain aspects of the internal model of the [Class-Feature-Centroid \(CFC\)](#) text classifier.*

Interactive Data Visualization for Generating Classifiers: This work proposes an application to generate classifiers from a visualization of the data. The information landscape embedded in an framework can be applied by domain experts to create classifiers from scratch (using arbitrary classification algorithms). The application has been deployed in two companies, however, as no extensive user study has been done, it can only be argued that users benefit from this kind of classifier creation. The results suggest, that *the information landscape as unsupervised data layout embedded in an interactive application allows domain experts to generate classifiers from scratch.*

Concept of Visual Active Learning: The concept of Visual Active Learning is proposed as an extension of classical active learning. Experiments using the developed classifier-agnostic visualizations show that classical active learning is – at least for the tested combinations of classifiers and data sets – outperformed by Visual Active Learning. This finding points toward the beneficialness of integrating the user in data mining processes. However, due to the complexity of the experimental space (data sets and classifier combinations, various runs for each combination), the findings are mostly based on simulations that model user behavior. We can only assume that experiments with trained users would lead to the same conclusions. The experiments compared Visual Active Learning and classical active learning in a classifier-independent way. We can not conclude that classical active learning strategies tailored towards classifiers are also outperformed by its visual pendant. The results suggests, that *Visual Active Learning with the Class Radial Visualization outperforms classical classifier-agnostic active learning and performs never worse than random sampling.*

Evaluation of Text Representations for Faster Labeling: For the purpose of minimizing the time required to generate training data for text classification key phrases and key sentences were identified as alternative text representation forms. This approach minimizes the time required to label single documents and can be easily combined with approaches minimizing the total amount of training data, like active learning. The experiments show, that the key phrases representation allows users to accurately and fast label training data. The user study was performed on a German data set for the task of topic detection. It is likely, that the key word extraction approach would have to be adapted for other tasks, e.g. sentiment detection. The results suggests, *that the key word representation allows users to accurately and fast label training data for the task of topic detection.*

6.2 Directions for Future Work

In this work we presented evidence that it is beneficial to include the user in the processes of generation and adaptation of classifiers using visualizations. On the one hand, better performing classifiers can be created more efficiently using domain knowledge of the user. On the other hand, users might get the chance to better understand and eventually trust the automatic methods. The main directions for future work that can be concluded from the discussion in the last section are the following:

This thesis focuses on single-label classification problems. Especially, text classification problems are inherently multi-label problems. Although multi-label problems can always be reduced to multiple single-label problems on the algorithmic side, it needs to be investigated how generalization can be achieved for visually supported classification.

In this work we only partially exploited the potential of visualization techniques. Further extensions include multiple coordinated views [WBWK00] combined with brushing and linking techniques. For example, the Confusion Map could be extended such that by clicking on a cell the corresponding items are shown – either in a pure data visualization or in highlighted in the Class Radial Visualization. This might support further insights to why some classes are confused by the algorithm.

Furthermore, the technique of lasso selection implemented in the information landscape can be applied to the Class Radial Visualization. Lasso selection would give users the possibility to select and move multiple items at once and thus, to faster generate labeled data. However, this would also imply an adaptation of the classifier generation strategies, because items belonging to different classes would likely be moved together. A natural possibility to address this issue would be the implementation of multiple-instance learning [Bab08] techniques.

Extended users studies would allow to (i) strengthen or extend the experimental findings, and (ii) assess usability aspects of single visualizations and the prototypical applications.

List of Abbreviations

ANN Artificial Neural Network.

BM25 Okapi BM25 ranking.

CCA Curvilinear Component Analysis.

CFC Class-Feature-Centroid.

EDA Electronic Design Automation.

FDP Force-Directed Placement.

FN False Negatives.

FP False Positives.

KNN k-Nearest Neighbor.

LDA Linear Discriminant Analysis.

MDS Multi-Dimensional Scaling.

MLP Mult-Layer Perceptron.

NB Naive Bayes.

PCA Principal Component Analysis.

POS Part-of-speech.

SVM Support Vector Machine.

List of Abbreviations

TF-IDF Term Frequency - Inverse Document Frequency.

TN True Negatives.

TP True Positives.

Bibliography

- [AEEK99] Mihael Ankerst, Christian Elsen, Martin Ester, and Hans-Peter Kriegel. Visual classification: an interactive approach to decision tree construction. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 392–396, New York, NY, USA, 1999. ACM. 48
- [AEK00] Mihael Ankerst, Martin Ester, and Hans-Peter Kriegel. Towards an effective cooperation of the user and the computer for classification. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining (KDD)*, pages 179–188, New York, NY, USA, 2000. ACM. 48
- [AK00] F. Aurenhammer and R. Klein. *Handbook on Computational Geometry*, chapter Voronoi Diagrams, pages 201–290. Elsevier Science Publishing, Amsterdam, December 2000. 35
- [AKA91] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Mach. Learn.*, 6(1):37–66, 1991. 21
- [Axe04] Stefan Axelsson. Combining a bayesian classifier with visualisation: Understanding the IDS. In *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 99–108. ACM Press, 2004. 49
- [Bab08] Boris Babenko. Multiple instance learning: Algorithms and applications. Research Exam, UCSD Computer Science and Engineering Department, 2008. 162
- [Bec98] Barry G. Becker. Research report: Visualizing decision table classifiers. In *Information Visualization*, Los Alamitos CA, 1998. IEEE Computer Society Press. 48
- [Ber83] J. Bertin. *Semiology of Graphics: Diagrams, Networks, Maps*. 1983. 78
- [BHK97] Peter N. Belhumeur, João P. Hespanha, and David Kriegman. Eigenfaces

Bibliography

- vs. Fisherfaces: recognition using class specific linear projection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):711–720, 1997. 103
- [BKS97] Barry Becker, Ron Kohavi, and Dan Sommerfield. Visualizing the simple bayesian classifier. In *KDD Workshop Issues in the Integration of Data Mining and Data Visualization*, 1997. 48
- [BM05] Remo Aslak Burkhard and Michael Meier. Tube map visualization: Evaluation of a novel knowledge visualization application for the transfer of knowledge in long-term projects. *Journal of Universal Computer Science*, 11(4):473–494, apr 2005. 16
- [BP09] Jason Baldridge and Alexis Palmer. How well does active learning actually work?: Time-based evaluation of cost-reduction strategies for language documentation. In *Proc. of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 296–305, Morristown, NJ, USA, 2009. Association for Computational Linguistics. 58
- [BZ05] Kai Bielenberg and Marc Zacher. Groups in social software: Utilizing tagging to integrate individual contexts for social navigation. Master’s thesis, Program of Digital Media, University of Bremen, 2005. 55, 57, 85
- [CCH01] Doina Caragea, Dianne Cook, and Vasant Honavar. Gaining insights into support vector machine pattern classifiers using projection-based tour methods. In *Proceedings of the KDD Conference*, pages 251–256, 2001. 48, 49, 50
- [CCH04] Dianne Cook, Doina Caragea, and Vasant Honavar. Visualization in classification problems. In *Proceedings of the COMPSTAT Symposium*, 2004. 49, 50
- [CCWH08] Doina Caragea, Dianne Cook, Hadley Wickham, and Vasant Honavar. Visual data mining. chapter Visual Methods for Examining SVM Classifiers, pages 136–153. Springer-Verlag, Berlin, Heidelberg, 2008. 2, 4, 50
- [CH67] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21 – 27, jan. 1967. 20
- [CL01] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>. 114, 121
- [CMS99] Stuart Card, Jock Mackinlay, and Ben Shneiderman, editors. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, California, USA, 1999. 16, 74, 79
- [CZ08] David Chodos and Osmar Zaiane. ARC-UI: Visualization Tool for As-

- sociative Classifiers. In *Proc. International Conference on Information Visualisation*, pages 296–301, July 2008. [50](#)
- [DA08] Banu Diri and Songul Albayrak. Visualization and analysis of classifiers performance in multi-class medical data. *Expert Systems with Applications*, 34(1):628 – 634, 2008. [2](#), [44](#), [45](#), [50](#)
- [dCF09] Andr de Carvalho and Alex Freitas. A tutorial on multi-label classification techniques. In Ajith Abraham, Aboul-Ella Hassanien, and Vclav Snel, editors, *Foundations of Computational Intelligence Volume 5*, volume 205 of *Studies in Computational Intelligence*, pages 177–195. Springer Berlin / Heidelberg, 2009. [18](#)
- [DHS00] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2 edition, November 2000. [23](#), [27](#)
- [DMM08] Gregory Druck, Gideon Mann, and Andrew McCallum. Learning from labeled features using generalized expectation criteria. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 595–602, New York, NY, USA, 2008. ACM. [51](#)
- [Dol07] Henrico Dolfing. A visual analytics framework for feature and classifier engineering. Master’s thesis, University of Konstanz, 2007. [44](#), [49](#)
- [DP97] Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Mach. Learn.*, 29(2-3):103–130, 1997. [22](#)
- [DPHS98] Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM '98: Proceedings of the seventh international conference on Information and knowledge management*, pages 148–155, New York, NY, USA, 1998. ACM Press. [23](#)
- [DT98] Robert P. W. Duin and David M.J. Tax. Classifier conditional posterior probabilities. In *SSPR '98/SPR '98: Proceedings of the Joint IAPR International Workshops on Advances in Pattern Recognition*, pages 611–619. Springer-Verlag, 1998. [24](#)
- [EB04] Martin J. Eppler and Remo A. Burkhard. Knowledge visualization towards a new discipline and its fields of application. ICA Working Paper 2/2004, University of Lugano, 2004. [16](#)
- [FA10] A. Frank and A. Asuncion. UCI machine learning repository, 2010. [80](#)
- [FCH⁺08] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-

Bibliography

- Jen Lin. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008. [107](#), [114](#), [121](#), [141](#)
- [FH03] Eibe Frank and Mark Hall. Visualizing class probability estimators. In *In Lecture Notes in Artificial Intelligence 2838*, pages 168–179. Springer, 2003. [2](#), [44](#), [48](#)
- [Fis36] Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals Eugenics*, 7:179–188, 1936. [98](#)
- [FPSS96] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17:37–54, 1996. [7](#), [10](#)
- [FS95] Ken Fishkin and Maureen C. Stone. Enhanced dynamic queries via movable filters. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 415–420, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co. [68](#)
- [FWG09] Raphael Fuchs, Jürgen Waser, and Meister Eduard Gröller. Visual human+machine learning. *Proc. Vis 09*, 15(6):1327–1334, October 2009. [50](#)
- [Gas09] Caroline Gasperin. Active learning for anaphora resolution. In *Proc. of the NAACL Workshop on Active Learning for Natural Language Processing (HLT)*, pages 1–8, Morristown, NJ, USA, 2009. Association for Computational Linguistics. [58](#)
- [GL10] Vishal Gupta and Gurpreet Lehal. A survey of text summarization extractive techniques. *Journal of Emerging Technologies in Web Intelligence*, 2(3), 2010. [52](#)
- [Gra72] Ronald L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Inf. Process. Lett.*, 1(4):132–133, 1972. [123](#)
- [Gra03] Micheal Granitzer. Hierarchical text classification using methods from machine learning. Master’s thesis, Graz University of Technology, October 2003. [24](#)
- [Gro06] Dennis P. Groth. Visualizing distributions and classification accuracy. volume 0, pages 389–394, Los Alamitos, CA, USA, 2006. IEEE Computer Society. [49](#)
- [GS07] Yuhong Guo and Dale Schuurmans. Discriminative batch mode active learning. In *Proc. of Advances in Neural Information Processing Systems (NIPS)*, 2007. [58](#)
- [GZG09] Hu Guan, Jingyu Zhou, and Minyi Guo. A class-feature-centroid classifier for text categorization. In *Proc. of the International conference on World Wide Web (WWW)*, pages 201–210, New York, NY, USA, 2009. ACM. [23](#), [24](#), [141](#)

- [Har68] P. Hart. The condensed nearest neighbor rule (corresp.). *Information Theory, IEEE Transactions on*, 14(3):515 – 516, may. 1968. [21](#)
- [HBP01] Ali Hadjarian, Jerzy Bala, and Peter Pachowicz. Text categorization through multistrategy learning and visualization. In *CICLing '01: Proceedings of the Second International Conference on Computational Linguistics and Intelligent Text Processing*, pages 437–443, London, UK, 2001. Springer-Verlag. [48](#)
- [Hea96] Christopher G. Healey. Choosing effective colours for data visualization. In *VIS '96: Proceedings of the 7th conference on Visualization '96*, pages 263–ff., Los Alamitos, CA, USA, 1996. IEEE Computer Society Press. [46](#), [73](#), [160](#)
- [HFH⁺09] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11:10–18, November 2009. [114](#), [141](#)
- [HK00] Eui-Hong Han and George Karypis. Centroid-based document classification: Analysis and experimental results. In *Proc. European Conf. on Principles of Data Mining and Knowledge Discovery (PKDD)*, pages 424–431, London, UK, 2000. Springer-Verlag. [107](#)
- [HK07] Martin Halvey and Mark T. Keane. An assessment of tag presentation techniques. In *Proceedings of the 16th International World Wide Web Conference (WWW2007)*, Banff, Alberta, Canada, May 2007. [82](#)
- [Hof99] Patrick Hoffman. *Table Visualizations: A Formal Model and its Applications*. PhD thesis, University of Massachusetts Lowell, 1999. [66](#)
- [Hof06] Kevin Hoffman. In search of the perfect tag cloud. Whitepaper, August 2006. [82](#), [85](#)
- [Joa98] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Heidelberg et al., 1998. Springer. [23](#)
- [JWR00] K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments. *Inf. Process. Manage.*, 36(6), 2000. [26](#), [108](#)
- [KHDM98] Josef Kittler, Mohamad Hatef, Robert P. W. Duin, and Jiri Matas. On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(3):226–239, 1998. [43](#)
- [KHGW07] Byron Y. L. Kuo, Thomas Hentrich, Benjamin M. Good, and Mark D. Wilkinson. Tag clouds for summarizing web search results. In *Proceed-*

Bibliography

- ings of the 16th International World Wide Web Conference (WWW2007)*, pages 1203–1204, New York, NY, USA, 2007. ACM Press. [55](#), [57](#)
- [KKEE11] KyungTae Kim, Sungahn Ko, N. Elmqvist, and D.S. Ebert. Wordbridge: Using composite tag clouds in node-link diagrams for visualizing content and relations in text corpora. In *System Sciences (HICSS), 2011 44th Hawaii International Conference on*, pages 1–8, jan. 2011. [55](#), [57](#)
- [KL07] Owen Kaser and Daniel Lemire. Tag-cloud drawing: Algorithms for cloud visualization. In *Proceedings of the Workshop on Tagging and Metadata for Social Information Organization (WWW2007)*, Banff, Alberta, Canada, May 2007. [55](#), [57](#)
- [KLM⁺00] Petri Kontkanen, Jussi Lahtinen, Petri Myllymäki, Tomi Silander, and Henry Tirri. Supervised model-based visualization of high-dimensional data. *Intell. Data Anal.*, 4(3,4):213–227, 2000. [2](#), [45](#), [48](#)
- [KMOZ08] Daniel A. Keim, Florian Mansmann, Daniela Oelke, and Hartmut Ziegler. Visual analytics: Combining automated discovery with interactive visualizations. In *Discovery Science (DS 2008)*, LNAI, pages 2–14, 2008. [16](#)
- [KMSZ06] Daniel A. Keim, Florian Mansmann, Jorn Schneidewind, and Hartmut Ziegler. Challenges in visual data analysis. In *IV '06: Proceedings of the conference on Information Visualization*, pages 9–16, Washington, DC, USA, 2006. IEEE Computer Society. [16](#)
- [Koh00] Ron Kohavi. Data mining and visualization. Invited talk at the National Academy of Engineering US Frontiers of Engineers (NAE), September 2000. [1](#)
- [Kot07] S. B. Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica*, 31(3):249–268, 2007. [18](#), [24](#), [103](#)
- [KP98] Ron Kohavi and Foster Provost. Glossary of terms. *Machine Learning*, 30(2-3):271–274, February 1998. [29](#)
- [KSM⁺09] Werner Klieber, Vedran Sabol, Markus Muhr, Roman Kern, and Michael Granitzer. Knowledge Discovery using the Knowminer framework. In *Proceedings of the IADIS International Conference on Information Systems*, pages 307–314, 2009. [148](#)
- [KT03] Ioannis Kopanakis and Babis Theodoulidis. Visual data mining modeling techniques for the visualization of mining outcomes. *Journal of Visual Languages & Computing*, 14(6):543 – 589, 2003. Visual Data Mining. [2](#), [49](#)
- [LC06] Loïc Lecerf and Boris Chidlovskii. Document annotation by active learning techniques. In *DocEng '06: Proceedings of the 2006 ACM symposium on*

- Document engineering*, pages 125–127, New York, NY, USA, 2006. ACM. [58](#)
- [LC09] Loïc Lecerf and Boris Chidlovskii. Visalix: A web application for visual data analysis and clustering. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2009. [45](#), [46](#), [50](#), [76](#), [77](#)
- [LSGJ09a] Elisabeth Lex, Christin Seifert, Michael Granitzer, and Andreas Juffinger. Automated blog classification: A cross-domain approach. In *Proc. of IADIS International Conference WWW/Internet*, 2009. [9](#), [100](#), [103](#)
- [LSGJ09b] Elisabeth Lex, Christin Seifert, Michael Granitzer, and Andreas Juffinger. Cross-domain classification: Trade-off between complexity and accuracy. In *Proceedings of the 4th International Conference for Internet Technology and Secured Transactions (ICITST)*, 2009. [9](#), [100](#), [103](#)
- [LSGJ10] Elisabeth Lex, Christin Seifert, Michael Granitzer, and Andreas Juffinger. Efficient cross-domain classification of weblogs. *International Journal of Computational Intelligence Research*, 1(1):73–82, 2010. [9](#), [78](#), [100](#), [103](#)
- [LSKG08] Elisabeth Lex, Christin Seifert, Wolfgang Kienreich, and Michael Granitzer. A generic framework for visualizing the news article domain and its application to real-world data. *Journal of Digital Information Management (JDIM)*, 6(6), Dec 2008. [82](#)
- [LT04] Verayuth Lertnattee and Thanaruk Theeramunkong. Effect of term distributions on centroid-based text categorization. *Information Sciences, Informatics and Computer Science*, 158(1), 2004. [107](#)
- [Mac67] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967. [148](#)
- [McC02] Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu>, 2002. [141](#)
- [MK08a] Thorsten May and Jörn Kohlhammer. Towards closing the analysis gap: Visual generation of decision supporting schemes from raw data. In *Joint Eurographics and IEEE VGTC Symposium on Visualization (EuroVis), Computer Graphics Forum*, volume 27, pages 911–918, 2008. [2](#), [7](#), [50](#)
- [MK08b] Thorsten May and Jörn Kohlhammer. Visual verification of hypotheses. In *ISVC '08: Proceedings of the 4th International Symposium on Advances in Visual Computing, Part II*, pages 31–42, Berlin, Heidelberg, 2008. Springer-Verlag. [50](#)
- [MP02] Ofer Melnik and Jordan B. Pollack. Theory and scope of exact representa-

Bibliography

- tion extraction from feed-forward networks. *Cognitive Systems Research*, 3(2):203 – 226, 2002. [48](#)
- [MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008. [26](#), [27](#)
- [MT04] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into texts. In *Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, 2004. [52](#), [132](#)
- [NNM96] Samer A. Nene, Shree K. Nayar, and Hiroshi Murase. Columbia object image library (coil-20). Technical Report CUCS-005-96, Department of Computer Science, Columbia University, February 1996. [98](#)
- [NSC06] Vivi Nastase, Jelber Sayyad Shirabad, and Maria Fernanda Caropreso. Using dependency relations for text classification. In *Proc. Nineteenth Canadian Conf. on Artificial Intelligence*, Quebec, Canada, 2006. [54](#)
- [OBSC00] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial Tesselations: Concepts and Applications of Voronoi Diagrams*. Probability and Statistics. John Wiley, NYC, 2nd edition, 2000. 671 pages. [35](#)
- [Pal02] W.B. Paley. Textarc: Showing word frequency and distribution in text. In *Proceedings of IEEE Symposium on Information Visualization, Poster Compendium*, IEEE CS Press, 2002. [52](#), [53](#)
- [PES⁺06] Brett Poulin, Roman Eisner, Duane Szafron, Paul Lu, Russ Greiner, D. S. Wishart, Alona Fyshe, Brandon Pearcy, Cam MacDonell, and John Anvik. Visual explanation of evidence in additive classifiers. In *Proc. Conference on Innovative Applications of Artificial Intelligence (IAAI06)*, pages 1822–1829, Boston, MA, July 2006. [44](#), [49](#)
- [PF97] Foster Provost and Tom Fawcett. Analysis and visualization of classifier performance with nonuniform class and cost distributions. In *Workshop on AI Approaches to Fraud Detection and Risk Management*, 1997. [44](#), [48](#)
- [Pla99] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999. [24](#)
- [Pou04] Francois Poulet. Svm and graphical algorithms: A cooperative approach. In *ICDM '04: Proceedings of the Fourth IEEE International Conference on Data Mining*, pages 499–502, Washington, DC, USA, 2004. IEEE Computer Society. [49](#), [50](#)
- [Pou08] François Poulet. *Towards Effective Visual Data Mining with Cooperative Approaches*, pages 389–406. Springer-Verlag, Berlin, Heidelberg, 2008. [50](#)

- [PRY⁺06] Catherine Plaisant, James Rose, Bei Yu, Loretta Auvil, Matthew G. Kirschenbaum, Martha Nell Smith, Tanya Clement, and Greg Lord. Exploring erotics in emily dickinson’s correspondence with text mining and visual interfaces. In *JCDL ’06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 141–150, New York, NY, USA, 2006. ACM. 49
- [Rd00] Penny Rheingans and Marie desJardins. Visualizing high-dimensional predictive model quality. In *Proceedings of IEEE Visualization*, pages 493–496, 2000. 2, 44, 48
- [RHC10] Troy Raeder, T. Ryan Hoens, and Nitesh V. Chawla. Consequences of variability in classifier performance estimates. In *International Conference on Data Mining*, pages 421–430, Sydney, Australia, Dec 2010. 28
- [Ros06] Fabrice Rossi. Visual data mining and machine learning. In *ESANN 2006, 14th European Symposium on Artificial Neural Networks*, pages 251–264, 2006. 17
- [SC00] Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *ICML ’00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 839–846, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. 58
- [Seb02] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002. 17, 21, 27, 29, 30, 107
- [Set10] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2010. 34, 51, 58, 60, 121, 126
- [SG10] Christin Seifert and Michael Granitzer. User-based active learning. In Wei Fan, Wynne Hsu, Geoffrey I. Webb, Bing Liu, Chengqi Zhang, Dimitrios Gunopulos, and Xindong Wu, editors, *Proceedings of 10th International Conference on Data Mining Workshops (ICDM2010)*, pages 418–425, Sydney, Australia, Dec 2010. 10, 58, 113
- [Sha05] Blake Shaw. Utilizing folksonomy: Similarity metadata from the del.icio.us system. Project Proposal, December 2005. 55, 57
- [Shn92] Ben Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics (TOG)*, 11(1):92–99, 1992. 15, 16
- [Shn02] Ben Shneiderman. Inventing discovery tools: combining information visualization with data mining. *Information Visualization*, 1(1):5–12, 2002. 2
- [SHZ⁺07] Yang Song, Jian Huang, Ding Zhou, Hongyuan Zha, and C. Lee Giles.

- Iknn: Informative k-nearest neighbor pattern classification. In *Proc. of the European conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 248–264, Berlin, Heidelberg, 2007. Springer-Verlag. [21](#), [114](#)
- [SJS06] Marina Sokolova, Nathalie Japkowicz, and Stan Szpakowicz. Beyond Accuracy, F-score and ROC: A family of discriminant measures for performance evaluation. In *Advances in Artificial Intelligence (AI2006)*, LNAI 4304, pages 1015–1021, Berlin/Heidelberg, 2006. Springer. [30](#)
- [SKG11] Christin Seifert, Wolfgang Kienreich, and Michael Granitzer. Visualizing text classification models with voronoi word clouds. In *Proceedings 15th International Conference Information Visualisation (IV)*, 2011. poster. [10](#), [91](#), [155](#)
- [SKK⁺08] Christin Seifert, Barbara Kump, Wolfgang Kienreich, Gisela Granitzer, and Michael Granitzer. On the beauty and usability of tag clouds. In *Proceedings of the 12th International Conference on Information Visualisation (IV)*, pages 17–25, Los Alamitos, CA, USA, July 2008. IEEE Computer Society. [10](#), [52](#), [55](#), [56](#), [82](#), [133](#)
- [SKM⁺09] Vedran Sabol, Wolfgang Kienreich, Markus Muhr, Werner Klieber, and Michael Granitzer. Visual knowledge discovery in dynamic enterprise text repositories. In *IV '09: Proceedings of the 2009 13th International Conference Information Visualisation*, pages 361–368, Washington, DC, USA, 2009. IEEE Computer Society. [9](#), [148](#)
- [SKVH09] Naeem Seliya, Taghi M. Khoshgoftaar, and Jason Van Hulse. Aggregating performance metrics for classifier evaluation. In *IRI'09: Proceedings of the 10th IEEE international conference on Information Reuse & Integration*, pages 35–40, Piscataway, NJ, USA, 2009. IEEE Press. [30](#)
- [SL09a] Christin Seifert and Elisabeth Lex. A novel visualization approach for data-mining-related classification. In *Proc. of the International Conference on Information Visualisation (IV)*, pages 490–495. Wiley, July 2009. [9](#), [65](#), [103](#)
- [SL09b] Christin Seifert and Elisabeth Lex. A visualization to investigate and give feedback to classifiers. In *Proceedings European Conference on Visualization (EuroVis)*, Jun 2009. poster. [9](#), [65](#)
- [SLG⁺03] Duane Szafron, Paul Lu, Russell Greiner, David Wishart, Zhiyong Lu, Brett Poulin, John Anvik, and Cam Macdonell. Proteome analyst transparent high-throughput protein annotation: function, localization and custom predictors. In *International Conference on Machine Learning Workshop on Machine Learning in Bioinformatics (ICML-Bioinformatics)*, 2003. [45](#), [49](#)

- [SLT09] Johann Schrammel, Michael Leitner, and Manfred Tscheligi. Semantically structured tag clouds: an empirical evaluation of clustered presentation approaches. In *Proceedings of the 27th international conference on Human factors in computing systems, CHI '09*, pages 2037–2040, New York, NY, USA, 2009. ACM. 56
- [SMPK06] Gregor Stiglic, Matej Mertik, Vili Podgorelec, and Peter Kokol. Using visual interpretation of small ensembles in microarray analysis. In *Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems*, pages 691–695, Washington, DC, USA, 2006. IEEE Computer Society. 49
- [SOR⁺09] Hendrik Strobelt, Daniela Oelke, Christian Rohrdantz, Andreas Stoffel, Daniel A. Keim, and Oliver Deussen. Document cards: A top trumps visualization for documents. *IEEE Transactions on Visualization and Computer Graphics*, 15:1145–1152, 2009. 52
- [Spe06] Robert Spence. *Information Visualization: Design for Interaction*. Number 0132065509. Prentice Hall, 2nd edition edition, 2006. 16
- [SSG10] Christin Seifert, Vedran Sabol, and Michael Granitzer. Classifier hypothesis generation using visual analysis methods. In Filip Zavoral, Jakob Yaghob, Pit Pichappan, and Eyas El-Qawasmeh, editors, *Networked Digital Technologies*, volume 87 of *Communications in Computer and Information Science*, pages 98–111. Springer, 2010. 9, 147
- [SSK⁺ar] Christin Seifert, Vedran Sabol, Wolfgang Kienreich, Elisabeth Lex, and Michael Granitzer. *Large Scale Data Analytics*, chapter Visual Analysis and Knowledge Discovery for Text. Springer, toAppear. 10
- [Ste07] Moritz Stefaner. Visual tools for the socio-semantic web. Master’s thesis, University of Applied Sciences Potsdam, June 2007. 55, 57
- [SU07] Andrew I. Schein and Lyle H. Ungar. Active learning for logistic regression: an evaluation. *Mach. Learn.*, 68(3):235–265, 2007. 58
- [SUG11] Christin Seifert, Eva Ulbrich, and Michael Granitzer. Word clouds for efficient document labeling. In *The Fourteenth International Conference on Discovery Science*, October 2011. 10, 131
- [TBD⁺01] Kurt Thearling, Barry Becker, Dennis DeCoste, William Mawby, Michel Pilote, and Dan Sommerfield. *Information Visualization in Data Mining and Knowledge Discovery*, chapter Visualizing data mining models, pages 205–222. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001. 1
- [TC05] James J. Thomas and Kristin A. Cook, editors. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Computer Society, 2005. 16

Bibliography

- [TK01] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, 2001. [121](#)
- [TLKT09] Justin Talbot, Bongshin Lee, Ashish Kapoor, and Desney S. Tan. Ensemblematrix: interactive visualization to support machine learning with multiple classifiers. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 1283–1292, New York, NY, USA, 2009. ACM. [45](#), [46](#), [47](#), [50](#)
- [TO09] Katrin Tomanek and Fredrik Olsson. A web survey on the use of active learning to support annotation of text data. In *Proc. of the NAACL Workshop on Active Learning for Natural Language Processing (HLT)*, pages 45–48, Morristown, NJ, USA, 2009. Association for Computational Linguistics. [58](#), [60](#)
- [TP91] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–86, 1991. [103](#)
- [uB10] Artur Šilić and Bojana Bašić. Visualization of text streams: A survey. In Rossitza Setchi, Ivan Jordanov, Robert Howlett, and Lakhmi Jain, editors, *Knowledge-Based and Intelligent Information and Engineering Systems*, volume 6277 of *Lecture Notes in Computer Science*, pages 31–43. Springer Berlin / Heidelberg, 2010. [52](#)
- [Urb02] Simon Urbanek. Exploring statistical forests. In *Proc. of the 2002 Joint Statistical Meeting*, 2002. [48](#)
- [Vap95] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995. [22](#)
- [Vap98] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 1998. [22](#)
- [vHWV09] Frank van Ham, Martin Wattenberg, and Fernanda B. Viegas. Mapping text with phrase nets. *IEEE Transactions on Visualization and Computer Graphics*, 15:1169–1176, November 2009. [16](#), [52](#), [53](#)
- [VWF09] Fernanda B. Viégas, Martin Wattenberg, and Jonathan Feinberg. Participatory visualization with wordle. *IEEE Transactions on Visualization and Computer Graphics*, 15:1137–1144, 2009. [52](#), [53](#), [55](#), [57](#), [133](#)
- [War04] Colin Ware. *Information Visualization - Perception for Design*. Morgan Kaufmann, 2004. [16](#)
- [WBWK00] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '00, pages 110–119, New York, NY, USA, 2000. ACM. [162](#)

- [WEH⁺01] Malcolm Ware, Frank Eibe, Geoffrey Holmes, Mark Hall, and Ian H. Witten. Interactive machine learning: letting users build classifiers. *International Journal of Human-Computer Studies*, 55(3):281 – 292, 2001. [2](#), [48](#)
- [WF05] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition edition, 2005. [31](#), [98](#), [103](#)
- [WF09] Leland Wilkinson and Michael Friendly. The history of the cluster heat map. *The American Statistician*, 63(2):179–184, May 2009. [78](#)
- [WKRQ⁺08] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey McLachlan, Angus Ng, Bing Liu, Philip Yu, Zhi-Hua Zhou, Michael Steinbach, David Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14:1–37, 2008. 10.1007/s10115-007-0114-2. [18](#)
- [WM97] David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on evolutionary computation*, 1(1):67–82, 1997. [24](#)
- [Wol96a] David H. Wolpert. The existence of a priori distinctions between learning algorithms. *Neural Comput.*, 8(7):1391–1420, 1996. [18](#), [27](#)
- [Wol96b] David H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390, 1996. [1](#)
- [WPL⁺11] Yingcai Wu, Thomas Provan, Shixia Liu, Furu Wei, and Kwan-Liu Ma. Semantic-preserving word cloud generation by seam carving. *Computer Graphics Forum*, 30(3), June 2011. (EuroVis 2011). [55](#), [57](#)
- [WT04] Pak Chung Wong and Jim Thomas. Visual analytics. *IEEE Computer Graphics and Applications*, 24:20–21, 2004. [17](#)
- [WV08] Martin Wattenberg and Fernanda B. Viégas. The word tree, an interactive visual concordance. *IEEE Transactions on Visualization and Computer Graphics*, 14:1221–1228, November 2008. [52](#), [53](#)
- [YL99] Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Proc. Int. ACM Conf. on Research and Development in Information Retrieval(SIGIR)*, pages 42–49, New York, NY, USA, 1999. ACM. [108](#)
- [Zhu08] Xiaojin Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin, 2008. [51](#)
- [ZK02] Ying Zhao and George Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 515–524, New York, NY, USA, 2002. ACM. [148](#)