Feature-Based Visual Exploration of Text Classification

Florian Stoffel[*] University of Konstanz
Lucie Flekova[†] Technical University of Darmstadt
Daniela Oelke[‡] Siemens AG
Iryna Gurevych[§] Technical University of Darmstadt
Daniel A. Keim[*] University of Konstanz

Abstract

There are many applications of text classification such as gender attribution in market research or the identification of forged product reviews on e-commerce sites. Although several automatic methods provide satisfying performance in most application cases, we see a gap in supporting the analyst to understand the results and derive knowledge for future application scenarios. In this paper, we present a visualization driven application that allows analysts to gain insight in text classification tasks such as sentiment detection or authorship attribution on feature level, built with a practitioner's way of reasoning in mind, the *Text Classification Analysis Process*.

1 Introduction

With the increased availability and rapid growth of textual data, analyzing text data has gained tremendous popularity. The huge variety of applications includes, but is not limited to, sentiment analysis, essay grading, user profiling, automated feedback processing, or the partitioning of a given document collection into various different topics. The foundation of these applications are machine learning techniques, which employ feature vectors extracted from the text data. These feature vectors are composed out of a number different features, for example statistics that measure text properties like average length of a sentence, or lexicon features which determine the occurrence or share of lexicon words in a given text document. Most text data applications can be implemented using freely available libraries like Stanford CoreNLP [16]. They do not require a high level of expertise in natural language processing, work reasonably well for most applications, and do not impose detailed knowledge of the actual feature set. However, having a text application, analysis on feature level can be very informative in order to understand common errors or flaws in the outcome of the machine learning methods, because besides word occurrences and statistical properties semantics play a role too. For example, "enjoy" is usually of very positive polarity, although a negation (*didn't*) can turn it to be negative: "Alice **didn't** enjoy riding Bobs new bike". Adding heuristics for this or similar cases is useful only to a limited extend, because they cannot include all possible variations of negations, as they are a linguistic phenomenon which are very volatile for various reasons. This also holds for a variety of other problems in natural language processing, for example the detection and proper handling of irony or sarcasm. The dynamics and semantics of natural language are one of the major reasons why working with text data is challenging. To cope with these different challenges, we propose that analysts visually inspect the feature set in order to get an idea of the cause of errors or unexpected outcomes that is visible on feature level, given that the technology used is working as expected. The formalization of this process is a six stage procedure which we call *Text Classification Analysis Process* (TeCAP) (see Figure 1), which has been developed in close collaboration with practitioners in the field of machine learning and natural language processing. TeCAP contains three phases, consisting of six stages: 1. **Initialization Phase**: the machine learning task is executed and the results are modeled (stage one). 2. **Exploration and Examination Phase**: exploration of machine learning results on feature level, observation and validation of findings (stages two to five). 3. **Design Phase**: insights from the previous phase can be used in order to change the feature set (stage six).

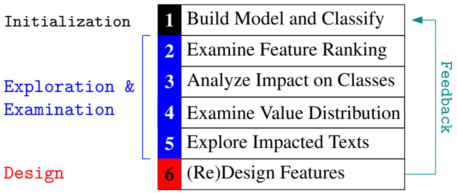

Initialization
Exploration &
Examination
Design
1 Build Model and Classify
2 Examine Feature Ranking
3 Analyze Impact on Classes
4 Examine Value Distribution
5 Explore Impacted Texts
6 (Re)Design Features
Feedback

Figure 1: Text Classification Analysis Process (TeCAP). It contains three phases, the *Initialization Phase* (black), the *Exploration and Examination Phase* (blue), and the *Design Phase* (red).

After the *Initialization Phase*, analysts are free to choose which visualization they use, although the level of detail on each stage varies from very high level (importance of features) down to the actual feature level (occurrences in text). To account for insights in the application problem, each of the stages can be skipped to reach the feedback loop from stage six to sage one.

In this paper, we claim the following contributions: 1. The structuring of a feature based machine learning exploration technique *TeCAP*. 2. The prototypical implementation of TeCAP in a standalone application *Minerva*, that allows the exploration of text classification results on feature level using visual analytics techniques. 3. We demonstrate the applicability and usefulness of Minerva on a real world problem in an application example.

2 Related Work

Although our work is not focusing on feature selection and visual applications of feature selection in particular, we have foundations in that discipline. Guyon and Eliseff [8] introduce different ranking and selection techniques, which are considered as standard today. An early work bringing together visualization and feature selection in an interactive manner has been published by Guo [6]. Based on the selection of subspaces in a high dimensional data space, interactive visualizations are provided in order to allow analysts to explore the data space. Noteworthy is the integration of steerable techniques to support the data exploration like orderings, groupings, and the control of aggregation methods. May et al. present a visualization technique designed for feature subset selection called *SmartStripes* [17]. The authors tightly integrate feature selection algorithms and visualization in order to allow the user to refine and steer the automatic feature selection. Krause et al. [11] present a system based on similar principles, but in contrast to *SmartStripes* it is designed to support predictive modeling in a specific use case.

[*] e-mail: {florian.stoffel,daniel.keim}@uni-konstanz.de
[†] e-mail: flekova@ukp.informatik.tu-darmstadt.de
[‡] e-mail: daniela.oelke@siemens.com; affiliation when the paper was written: German Institute for International Educational Research (DIPF)
[§] e-mail: gurevych@ukp.informatik.tu-darmstadt.de; also affiliated with German Institute for International Educational Research (DIPF)

To do so, a specific glyph design and ranked layouts of them are applied.

Application wise, Mayfield and Penstein-Rosé are closely related to our work [18]. They report on an interactive application designed to support error analysis in text classification tasks based on a matrix display of the confusion matrix. Heimerl et al. introduce a system which combines instance level visualization of the classification and a cluster view [10]. A cluster exploration system for linguistically motivated data is introduced by Lamprecht et al. in [13]. Seifert et al. propose a user-driven classification process by visualizing the classifier confidence and input documents [23]. Ankerst et al. visualize features, but in contrast to our system work only with decision trees [1]. A similar application is presented by van den Elzen and van Wijk [29]. Seo and Shneiderman present a system implementing a rank-by-feature framework [24]. They use multiple visualizations such as matrices, histograms, and scatter plots to visualize the features and various statistics.

There has already been work on reasoning of feature combinations and selections in machine learning tasks [30, 15]. This aspect of machine learning in the text application domain is the main motivation for us to add the design phase to TeCAP.

The related work shows that there has been very little work to provide the ability to analyze applied methods on feature level, which is in our understanding required to understand the outcome of text mining, because of the aforementioned inherent semantic dimension and dynamics of natural language text data. This is the gap we want to bridge with TeCAP and Minerva.

3 MINERVA

The prototypical implementation of TeCAP is called Minerva. It supports the *Exploration and Examination Phase* with visualizations and includes facilities supporting the *Design Phase*. In the following, we outline the system and present our visualization designs for each of the stages in the *Exploration and Examination Phase*.

3.1 System Design

Minerva has five main components, which are: 1. Input (load feature vectors); 2. Classification Model Creation (input of classes and confusion matrix); 3. Data Processing (filter, order, combine, remove); 4. Visualization; 5. Data Export (export feature vectors). Each component operates on separate input data, which allows the examination of different data sources at the same time, for example to compare the outcome of two different feature sets extracted from the same data set.

The design abstracts from specific machine learning libraries or applications in order to allow the examination of different machine learning techniques or feature sets on the same data. The system *reads the feature vectors* from ARFF files, as produced by WEKA [9] and similar libraries, CSV, and text files. The *classification model* allows Minerva to examine the machine learning algorithm outcome in detail. This includes the ability to judge whether a data instance has been misclassified and, if a confusion matrix has been provided, whether a misclassified instance belongs to false negatives or true negatives. The *processing* component provides utilities for the visualization (filtering, ordering), as well as standalone functionality to combine or remove features (design stage of TeCAP). The result is seamlessly integrated in the data model, so that any connected visualization or export component uses the ordered, filtered, or created features together with the imported ones.

Changes in the feature set can be stored in ARFF files that can be used as input for the popular machine learning library WEKA. The *export facility* allows filtered or combined features in the output, making it a suitable mechanism to re-run the machine learning task in order to inspect any differences afterwards.

3.2 Visualization Designs

Stage 2: Examine Feature Ranking. To give a quick impression of the importance of a feature in a potentially very large feature set (in our experiments we used feature sets with more than 5,000 features), we provide a word cloud of the top *n* most important features. Limiting the numbers and also the possibility of restricting or excluding feature labels ensures that the analysts has tools at hand to reduce the amount of displayed features to specific features at hand, while at the same time provides capabilities of a quick overview of the whole feature set. Feature importance is double encoded in the label size and color. This makes sure that even if the label of an important feature is shorter than others the analyst is still able to perceive the feature as important, and the label is not getting lost in the word cloud. The importance of features is computed according to state of the art measures such as information gain, symmetrical uncertainty, or the chi-square test statistic.

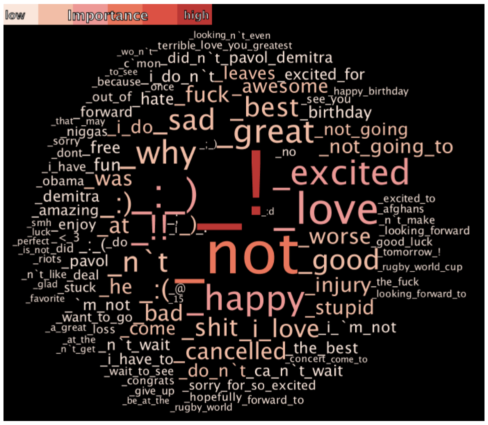


Figure 2: Word cloud displaying 100 features and their importance computed by the chi-square test statistic. The importance of a feature is double encoded in its color and the size of the feature label.

Analysts are able to adjust the number of displayed features as well as filter features or feature families for ex- or inclusion in the visualization. The word cloud layout is based on Rolled-out Wordles from Strobelt et al. [27], as it can be seen in Figure 2, which is known to generate compact layouts suitable for interactive systems.

Stage 3: Analyze Impact on Classes. At this stage, analysts can examine which features have predictive power for which class. For each feature, a glyph is displayed, which is built out of four equal sized segments of a circle (see Figure 3). The design is inspired by Guyon and Eliseff [8], as they show that features with distinct properties, like a distinct value distribution per instance set, can be used to form more predictive ones.

For each data instance, we compute the average value of true positives and true negatives to false negatives and false positives respectively. Each of the two segments showing the difference to the average false negatives/false positives is mapped with the same color. The visual design leads to four distinct error patterns which give a good idea of why a classification error has occurred (see Figure 4). The patterns are: 1. Features where the average value of false negative instances is closer to true negatives than true positives. They are likely to cause false negatives. 2. Features where the average value of false positive instances is closer to true positives than to true negatives. They are likely to cause false positives. 3. Features where both, 1 and 2 are the case. Those features are a potential cause of both, false negatives and false positives. 4. Features where both, 1 and 2 are not the case. This indicates good predictive power of the feature on misclassified instances.

Besides the visible ordering based in information gain, we also implemented a glyph ranking based on its visual properties, for

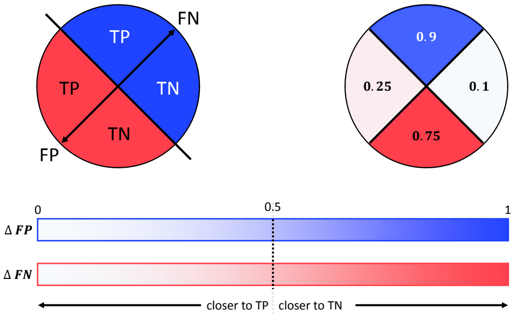


Figure 3: Class impact visualization. Top left shows the segmentation and the corresponding reference values. Read: TP and FN as $\Delta(\overline{TP} - \overline{FN})$ (top segment). On the top right, an example with the given values is shown. The color maps used in each sector are shown at the bottom.

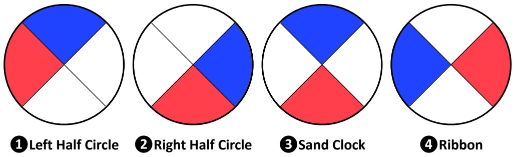


Figure 4: The four error patterns. Red sectors indicate the difference to false negatives, cyan sectors the difference to false positives. White colored segments do not contribute to the described patterns and are therefore left blank.

example error pattern affinity, in order to make it easy to spot groups of similarly behaving features. A visualization of a whole feature family (part of speech tags and part of speech tag patterns) can be seen in Figure 5. To enable the comparison of different features, a normalization relative to the minimal and maximal average true positive and average true negative value is applied for each glyph separately. The resulting values of misclassified instances can be inspected relative to correct classified instances.

The visualization makes two simplifying assumptions to make sure the visual design reflects the desired properties of a feature. i) The distribution of correct and incorrect classified instances has roughly the same shape. ii) The peak of the distribution of misclassified instances lies between the distribution peaks of correctly classified instances. The validity of these two assumptions can be verified with more detailed visualizations provided by Minerva.

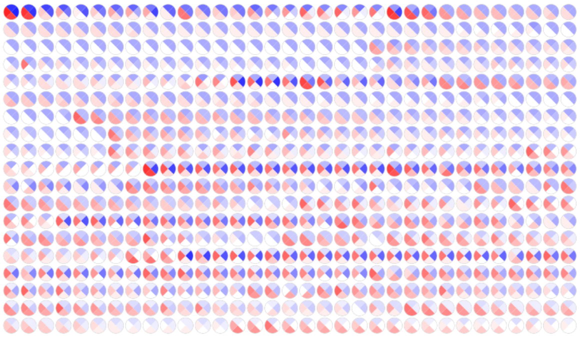


Figure 5: Visualization of relative feature value differences between the four instance classes. "Half-circle" glyphs indicate features which are likely to cause false positives or false negatives. The order of glyphs is determined by the displayed visual patterns.

Stage 4: Examine Value Distribution. At this stage, analysts can examine the distribution of feature values and also get information about the size of the overlap in different classes. To show this distribution, we combine two classes from the classification in a histogram display, as it can be seen in Figure 6, which enables comparison of the value distribution of two classes, for example false negatives and false positives. The height and background color of a histogram bar reflects the class with the most instances in the corresponding bin, the class with the smaller number of instances is indicated by the color and height of the inner T.

If necessary, the analyst can enable an additional coloring of the remaining background space of a histogram bin, which indicates the total number of instances in a bin with a color scale from dark gray (fewest) to white (most). This explicitly states the number of instances in the bin and allows the bin-wise comparison of number of instances not only for one feature, but also the complete feature set. See Figure 9 for an illustration of that feature.

Stage 5: Explore Impacted Texts. The different visualizations presented in this section support the analyst in developing new hypotheses and to select interesting documents for error analysis. Together with the feature set, the classified text documents are the ultimate tool to confirm or falsify the hypothesis of an error source, because nothing is able to illustrate the outcome of a feature extractor better than the actual data source.

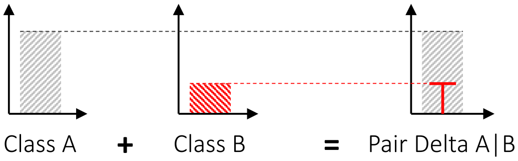
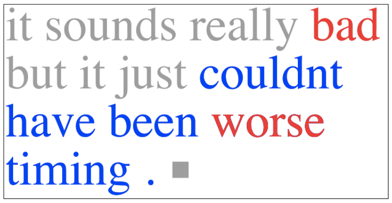


Figure 6: Pair Delta Visualization Construction. The histogram on the right is created by overlaying the two histograms on the left. The class with the most instances in a bin is represented by the bar color on the right, the smaller class is indicated by the color of the inner T.



it sounds really bad but it just couldnt have been worse timing . ■

Figure 7: Document Viewer. A tweet with highlighted polar words (red: negative polarity) and negation spans (blue). In this example, the tweet's polarity score was computed to be 0, although the message is clearly of negative nature.

To allow analysts verify or falsify their hypothesis, Minerva implements a document view which is augmented by the extracted features. The view allows the visualization of feature families like n-grams, negations, modal verbs, or word endings, based on the imported feature set. Furthermore, custom lexicons can be added if required by the analyst. Selected feature families are highlighted directly in the text by coloring the text span corresponding to the features (see Figure 7).

3.3 Interactive Visualizations

Minerva provides a general framework for interactive visualizations, which is mandatory for each of the different views. It is based on an infinite canvas and provides zooming and panning capabilities in order to facilitate the Visualization Information-Seeking Mantra by Shneiderman: *Overview first, zoom and filter, then details-on-demand* [25].

To foster the combination of different views for an effective exploration and examination of the data, Minerva provides a linking and brushing functionality [2]. Each visualization implements brushing mechanism suitable to the visual mapping of the features and propagates selection and de-selections of features to the selection subsystem, which in turn notifies the remainder of the system about changes of the selected features.

Besides the linking and brushing functionality, Minerva provides a view synchronization facility. This is realized by describing the current viewport of a visualization in terms of the displayed features. The abstraction from the graphical contents of a view makes it possible to synchronize the viewports of different kind of views. View synchronization can be enabled and disabled by the analyst, which makes Minerva suitable for explorative analysis as well as hypothesis building and verification tasks.

The combination of these three functionalities – linking, brushing, and view synchronization – allow analysts to switch visualizations and walk through TeCAP while maintaining focus at the currently selected feature set.

4 Application Example

In this section, we show how Minerva can be used to gain knowledge about a machine learning tasks working with text data. We demonstrate a popular sentiment polarity detection task, using a publicly available dataset with Twitter data from the 8th International Workshop on Semantic Evaluations (SemEval 2014 Task 9, Subtask B). Our goal is to demonstrate that achieving better performance is possible also through better understanding – enabled by TeCAP – of the textual features rather than standard machine learning customization.

Stage 1: Initialization – Build Model and Classify Our goal is to determine whether a given tweet is of positive, neutral, or negative sentiment. We use WEKAs SVM-SMO classifier with the information feature selection filter. Feature extraction is done by the UIMA-based [4] open source DKPro framework [7]. The feature set is based on successful applications from the literature. It contains a number of word- and character-level n-grams [3], text surface properties such as interpunction [20] or smileys [12], sentiment lexicons [14, 19, 21, 26], and syntactic measures of individual part of speech tag ratios and groups (n-grams on part of speech level).

Minerva abstracts from the actual machine learning task in order to not depend on a single machine learning library and keep the applicability of our methods as general as possible. As a consequence, it is required that after loading the data an in-place classification model are configured by the analyst. This makes sure that the classification outcome and details about the classes are available in the application, despite the fact that the actual classification process runs outside Minerva.

Stage 2: Exploration and Examination – Examine Feature Ranking In practice, one of the most interesting questions with respect to a machine learning task and the feature set is: *What are the most useful features?* In Figure 2, the top 100 n-grams from the positive and negative sentiment classification output can be seen. The importance of smileys, swearwords, and interpunction is clearly visible, which is an indicator for designing and adding new features in that areas to the feature set.

Furthermore, not only n-grams but any other feature subgroups can be examined with this visualization. Our feature set contains LIWC lexicons [21], which are helpful to separate neutral tweets from emotional ones (LIWC is an analytical framework frequently used by psychologists). Figure 8 illustrates the importance of LIWC features in the classification task. Besides the expected influence of positive and negative emotion words, `Affect` and `Anger`, the frequency of personal pronouns `Ppron` (I, them, her) and verbs `Verbs` (walk, go, see) play an important role. It is also interesting to see, that the frequencies of assents (*Agree*, *OK*, *yes*, ...) and negations (*no*, *not*, *never*) are also important.

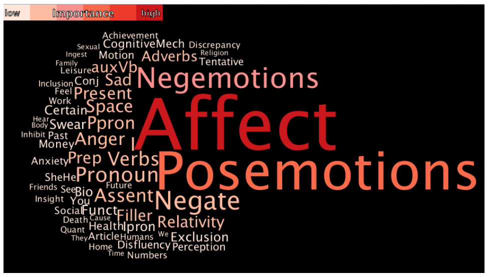


Figure 8: Word cloud of LIWC features and their importance in the sentiment distinction task. As expected, positive and negative emotion words have a large influence on the result and are therefore important. But it is also clearly visible, that the frequency of personal pronouns `Ppron` and verbs plays an important role.

Stage 3: Exploration and Examination – Analyze Impact on Classes The previous stage gives an initial understanding of which features matter in the sentiment polarity problem. Having now identified the important features, it is of interest to see which features have predictive power for which class.

Figure 5 shows features from Steinberger's polarity lexicon [26]. Each glyph represents a word from the polarity lexicon. Features corresponding to the left-half circle pattern (as introduced in Figure 4), are part of correct classification outcomes, if the represented word is present in a tweet, without causing false positives. If not, they lead to opposite conclusions in some cases, which results in false negatives. An opposite situation appears for the 5th feature in the top line, the *Ratio of verbs in tweet*. It suggests that in our test set a certain (low) verb rate predicts well a neutral tweet, while the other (high) verb rate cannot, on average, distinguish a polar tweet from a neutral one. Similarly, the positive-negative tweet problem can be analyzed. We observe that the right-half circle features are represented by n-grams such as *shit* or word groups such as *Disgust*, while left-half circles are lexicon words such as *Joy* or *n-grams* such as *looking forward*. Combining the left-half circle and right-half circle features (e.g., Joy-Disgust) in the preprocessing can lead to improved results, and at the same time eliminates the need for the demonstrated in-domain knowledge.

As just shown, the visualization is a useful instrument in order to refactor existing lexicons or create new ones, especially in tasks where the relation between the class and the words from the lexicon are not as clear as they are in the presented application.

Stage 4: Exploration and Examination – Examine Value Distribution With the help of the previously shown visualization, we were able to observe that numerous sentiment lexicons suffered from the same issue of predicting a polar tweet to be neutral when no lexicon word was found. What is missing is insight in the actual distribution of feature values and also information about the size of the overlap in different classes. In the sentiment classification task, the *Pair Delta Visualization* can be used to examine the problem that sentiment lexicon features perform badly when predicting a polar tweet to be neutral when the lexicon word is not part of the tweet to classify.

By the left and middle column of Figure 9, it becomes apparent, that even the combination of lexical features cannot lead to an improved classification performance of tweets with lexicon polarity value around zero. The highlighting in the background indicates, that close to zero values of the polarity are coming from other lexicon features as well. A possible explanation would be, that people indicate emotions without using sentiment words. However, for syntactic features (right column in Figure 9) the feature values are well distributed across value intervals, which makes a separation into two classes possible. Hence, combining syntactic features with the ones based on lexicon words could lead to a classification improvement.

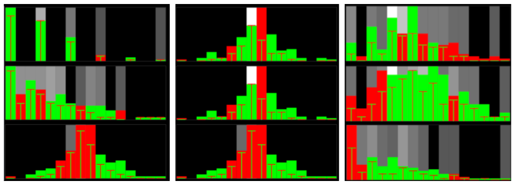


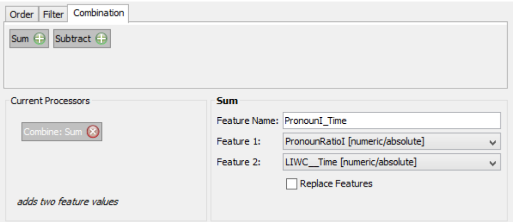
Figure 9: Value distribution of six sentiment lexicon features (left and middle column). While these lexical features share similar error overlap, impacted instances are distributed more evenly over syntactic feature values, as they can be seen in the right column.

Stage 5: Exploration and Examination – Explore Impacted Texts The exploration of impacted text can be used in order to see if *the feature computations have been well defined*. In particular, during this stage documents can be explored in order to find typical errors in misclassified documents, which could indicate that the feature measures a different phenomenon than intended.

In Figure 7, the resulting highlighting of polar word negation spans is illustrated. Using this visualization, we saw that while for certain words inverting the polarity score in negation was sensible (*"It doesn't sound bad"*, *"I wouldn't say it's great"*), for many cases it was counterproductive. The tweet shown in Figure 7 has a neutral polarity, because *bad* counts as `-1`, and *couldn't* + *worse* as `-(-1)`, which results in an overall polarity of `0`. Using this view, we also found out that skip-n-grams were not suitable features as they were ignoring occasional occurring negation words in between. Other errors come from ambivalent words such as loose (control vs. weight), or from ironic or sarcastic messages: *"now that I can finally sleep... can't wait to work for another 8 hours or so tomorrow... yay..."*.

Stage 6: Design – (Re)Design Features The last stage of TeCAP can be seen as the implementation from insights gained in the exploration and examination phase. In the spirit of Guyon [8], we allow feature combinations to be created directly in Minerva by providing an interface to create linear combination of existing features.

Using our built-in feature design facilities shown in Figure 10, we first combined positive and negative n-grams into features which behave as a sentiment lexicon. Additionally, we created combinations of all lexicon based features with syntactic features, especially verbs, pronouns, and adjective indicators.



Sum

Feature Name: PronounI_Time

Feature 1: PronounRatioI [numeric/absolute]

Feature 2: LIWC__Time [numeric/absolute]

□ Replace Features

adds two feature values

Figure 10: The user interface to create feature combinations. In this example, the combination of a semantic (time indicating words, LIWC) and a syntactic (pronoun ratio) feature is shown.

Lessons Learned and Results Based on the insights gained from the shown exploration and examination part of the Feature Engineering and Error Analysis Cycle, we adjusted the classification process as follows: 1. We added an additional sentiment lexicon based on positive and negative n-grams in order to enhance the existing polarity lexicons. 2. We combined lexicon based semantic and syntactic features, especially for verbs, pronouns, and adjective indicators. 3. The ArkTweet POS Tagger [5] has been complemented with the finer grained Stanford POS Tagger [28] in order to enhance the overall POS tagging accuracy. 4. The negation scoring was modified so that *"Can't be better"* is treated as positive and *"Can't be good"* as negative.

Besides the described run of the Feature Engineering and Analysis Cycle, further applications of the cycle and implementation of the suggested changes lead an improvement of the macro average F-Score from 56.2 to 64.1. This would place us in the final ranking of Semeval 2014 in the Top 20 of 50 participants, compared to the 38th rank we would reach with the initial setup without applying our analysis method (see `http://alt.qcri.org/semeval2014/task9/`).

5 Discussion

Semantic properties are a common cause for problems which can lead to a below par performance when applying machine learning techniques in natural language based application. Starting with this general problem, we designed the strategy *TeCAP* that conveys these problems to the human, who has a much broader knowledge of semantics and understands the analyzed text data. The strategy, developed in close collaboration with practitioners from the field of machine learning and natural language processing, gives the process of knowledge generation in text mining a structured way that can be followed easily and answers the most pressing questions when it comes to feature based analysis of machine learning outcomes.

For example, using the presented Feature Cloud users are able to quickly get an impression of the feature ranking. This information can be used to match users expectations to the actual machine learning tasks by confirming or falsifying previous knowledge of the data analyst.

The visualization for the impact analysis of features on the classes (Figure 5) is designed for specific error patterns. We abstract more from the actual feature vector data, but we still allow single feature analysis. The presented visual design distributes the feature glyphs in a grid, with customizable number of rows and columns. To improve the visual design in order to allow also the perception of clusters of features, we are currently working on an improved version of the visualization layout. When users want to focus more on the different groups (clusters) behaving similar in terms of caused errors, we plan to integrate a force directed layout instead of the currently available matrix view. To do so, we fix the four error patterns (Figure 4) in the edges of a square or rectangle, and place the single glyphs according to the attraction to these patterns. The resulting view should be able to effectively communicate the different groups in the feature set with respect to the error patterns. Having such a layout in place, new challenges rise, for example how to reduce the inevitable glyph overplotting. Since the feature ranking is very important for our partners, it would also be of interest to develop or apply existing techniques to include the ranking in the resulting visualization.

We also see potential for improvement in the text visualization, as it is shown in Figure 7. Currently, we use colored text spans, and lines above and under text spans to indicate where a feature is located in the text. This is complemented with a tool tip containing the names of features if they overlap at parts of text spans. We are examining text highlighting methods, for example background shadows or different font styles in order to be able to display more than three features at once and also visualize the spans where they overlap.

Another aspect we are working on is the integration of the different loops of visual analytics applications as proposed by the model of Sacha et al. [22]. There is huge potential for a *good* visual analytics based application in terms of the model, because the overall goal of our technique is gaining knowledge using insights from the feature level. We already have strong support for the exploration loop, and verification is possible because we provide different views on the same data. Including knowledge generation support would be a huge challenge, but a big achievement for data analysts in the field of natural language processing. In the current version of Minerva and the application domain of natural language processing, we rely heavily on world knowledge, which is hard to externalize and even harder to grasp via automatic processes. The first starting point to orientate our prototype more in the direction of the knowledge generation loop in terms of the model of Sacha et al. would be (semi)-automatic support for provenience of user actions like feature (de)-selection or combination.

6 Conclusion and Future Work

In this paper, we presented the Text Classification Analysis Process which has been developed in collaboration with practitioners in natural language processing and machine learning. We showcased a prototypical implementation tailored to text classification tasks and demonstrated, that the application TeCAP leads to insight with respect to the feature set and also improved the outcome in the application example.

We want to extend our work by exploring the design space of the visualizations, and enrich them with further aggregation and more advanced visualization techniques. In addition, it would be very interesting to investigate if TeCAP is general enough to fit application scenarios other as the one shown in this paper.

References

[1] M. Ankerst, C. Elsen, M. Ester, and H. Kriegel. Visual Classification: An Interactive Approach to Decision Tree Construction. In U. M. Fayyad, S. Chaudhuri, and D. Madigan, editors, *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 392–396. ACM, 1999.

[2] A. Buja, J. A. McDonald, J. Michalak, and W. Stuetzle. Interactive data visualization using focusing and linking. In *IEEE Visualization*, pages 156–163, 1991.

[3] W. B. Cavnar and J. M. Trenkle. N-Gram-Based Text Categorization. In *Proceedings of SDAIR-94, Third Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.

[4] D. A. Ferrucci and A. Lally. UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. *Journal of Natural Language Engineering*, 10(3-4):327–348, 2004.

[5] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (Short Papers)*, pages 42–47, 2011.

[6] D. Guo. Coordinating computational and visual approaches for interactive feature selection and multivariate clustering. *Information Visualization*, 2(4):232–246, 2003.

[7] I. Gurevych, M. Mühlhäuser, C. Müller, J. Steimle, M. Weimer, and T. Zesch. Darmstadt Knowledge Processing Repository Based on UIMA. In *Proceedings of the First Workshop on Unstructured Information Management Architecture at Biannual Conference of the GSCL*, 2007.

[8] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[9] M. A. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[10] F. Heimerl, C. Jochim, S. Koch, and T. Ertl. FeatureForge: A Novel Tool for Visually Supported Feature Engineering and Corpus Revision. In M. Kay and C. Boitet, editors, *COLING, (Posters)*, pages 461–470. Indian Institute of Technology Bombay, 2012.

[11] J. Krause, A. Perer, and E. Bertini. INFUSE: interactive feature selection for predictive modeling of high dimensional data. *IEEE Trans. Vis. Comput. Graph.*, 20(12):1614–1623, 2014.

[12] G. Laboreiro, L. Sarmento, J. Teixeira, and E. Oliveira. Tokenizing micro-blogging messages using a text classification approach. In R. Basili, D. P. Lopresti, C. Ringlstetter, S. Roy, K. U. Schulz, and L. V. Subramaniam, editors, *Proceedings of the Fourth Workshop on Analytics for Noisy Unstructured Text Data, AND 2010)*, pages 81–88. ACM, 2010.

[13] A. Lamprecht, A. Hautli, C. Rohrdantz, and T. Bögel. A Visual Analytics System for Cluster Exploration. In *51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, Proceedings of the Conference System Demonstrations*, pages 109–114. The Association for Computer Linguistics, 2013.

[14] B. Liu. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2012.

[15] Z. Liu. A comparative study on linguistic feature selection in sentiment polarity classification. *CoRR*, abs/1311.0833, 2013.

[16] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, System Demonstrations*, pages 55–60, 2014.

[17] T. May, A. Bannach, J. Davey, T. Ruppert, and J. Kohlhammer. Guiding feature subset selection with an interactive visualization. In *2011 IEEE Conference on Visual Analytics Science and Technology, VAST 2011*, pages 111–120, 2011.

[18] E. Mayfield and C. P. Rosé. An interactive tool for supporting error analysis for text mining. In *Proceedings of the NAACL HLT 2010 Demonstration Session*, pages 25–28. The Association for Computational Linguistics, 2010.

[19] S. Mohammad and P. D. Turney. Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence*, 29(3):436–465, 2013.

[20] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[21] J. W. Pennebaker, M. E. Francis, and R. J. Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71:2001, 2001.

[22] D. Sacha, A. Stoffel, F. Stoffel, B. C. Kwon, G. P. Ellis, and D. A. Keim. Knowledge generation model for visual analytics. *IEEE Trans. Vis. Comput. Graph.*, 20(12):1604–1613, 2014.

[23] C. Seifert, V. Sabol, and M. Granitzer. Classifier Hypothesis Generation Using Visual Analysis Methods. In *Networked Digital Technologies - Second International Conference, NDT 2010, Part I*, pages 98–111, 2010.

[24] J. Seo and B. Shneiderman. A Rank-by-Feature Framework for Interactive Exploration of Multidimensional Data. *Information Visualization*, 4(2):96–113, 2005.

[25] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *VL*, pages 336–343, 1996.

[26] J. Steinberger, M. Ebrahim, M. Ehrmann, A. Hurriyetoglu, M. A. Kabadjov, P. Lenkova, R. Steinberger, H. Tanev, S. Vázquez, and V. Zavarella. Creating sentiment dictionaries via triangulation. *Decision Support Systems*, 53(4):689–694, 2012.

[27] H. Strobelt, M. Spicker, A. Stoffel, D. A. Keim, and O. Deussen. Rolled-out Wordles: A Heuristic Method for Overlap Removal of 2D Data Representatives. *Comput. Graph. Forum*, 31(3):1135–1144, 2012.

[28] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT-NAACL*, 2003.

[29] S. van den Elzen and J. J. van Wijk. BaobabView: Interactive Construction and Analysis of Decision Trees. In *2011 IEEE Conference on Visual Analytics Science and Technology, VAST 2011*, pages 151–160, 2011.

[30] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.