A Machine Learning Approach to Sentiment Analysis in Multilingual Web Texts

Erik Boiy
erik.boiy@cs.kuleuven.be
Department of Computer Science
Katholieke Universiteit Leuven, Belgium

Marie-Francine Moens
sien.moens@cs.kuleuven.be
Department of Computer Science
Katholieke Universiteit Leuven, Belgium

August 13, 2008

Abstract

Sentiment analysis, also called opinion mining, is a form of information extraction from text of growing research and commercial interest. In this paper we present our machine learning experiments with regard to sentiment analysis in blog, review and forum texts found on the World Wide Web and written in English, Dutch and French. We train from a set of example sentences or statements that are manually annotated as positive, negative or neutral with regard to a certain entity. We are interested in the feelings that people express with regard to certain consumption products. We learn and evaluate several classification models that can be configured in a cascaded pipeline. We have to deal with several problems, being the noisy character of the input texts, the attribution of the sentiment to a particular entity and the small size of the training set. We succeed to identify positive, negative and neutral feelings to the entity under consideration with ca. 83% accuracy for English texts based on unigram features augmented with linguistic features. The accuracy results of processing the Dutch and French texts are ca. 70% and 68% respectively due to the larger variety of the linguistic expressions that more often diverge from standard language, thus demanding more training patterns. In addition, our experiments give us insights into the portability of the learned models across domains and languages. A substantial part of the article investigates the role of active learning techniques for reducing the number of examples to be manually annotated.

Keywords Opinion mining – information tracking – cross-language learning – active learning

1 Introduction

Automatic sentiment analysis regards the extraction of a sentiment from an unstructured source such as text, images or audio. The recognized sentiments can be classified as positive or negative, or a more fine grained sentiment classification scheme can be used. Sentiment analysis of text, also called opinion mining, only recently received a large interest from the academic community and commercial companies. What people write on persons, products or institutions has an important value in our society and the World Wide Web is an excellent source of such information.

The automatic analysis of sentiments on data found on the World Wide Web is useful for any company or institution caring about quality control. For the moment, getting user feedback means bothering him or her with surveys on every aspect the company is interested in. Making a survey for each product or feature, designing the format, distribution and timing of the survey (sending a form right after purchase might not be very informative), and the reliance on the goodwill of people to take the survey are expensive and time-consuming tasks, yielding not always accurate results. Surveying by means of questionnaires can be made obsolete by gathering such information automatically from the World Wide Web. One of the sources are blogs (short for "web logs"), a

medium through which the blog owner makes commentaries about a certain subject or talks about his or her personal experiences, inviting readers to provide their own comments. Other sources are customer review sites and electronic discussion boards or forums, where people can discuss all kinds of topics, or ask for other people's opinions.

There are several additional *advantages* to automated sentiment analysis. First, the people who share their views usually have more pronounced opinions than average. These opinions are additionally influencing others reading them, leading to so-called word-of-mouth marketing. Extracting these opinions is thus extra valuable. Second, opinions are extracted in real-time, allowing for quicker response times to market changes and for detailed time-based statistics that make it possible to plot trends over time. Last, but not least in information retrieval opinion mining assists in discriminating opinionated documents from documents that present the information in a neutral way.

This article presents our experiments with regard to sentiment analysis in *blog, review and forum texts* found on the World Wide Web and written in English, Dutch and French. We are interested in the feelings that people express with regard to certain consumption products. We learn several classification models from a set of examples that are manually annotated, more specifically, from sentences that are annotated as positive, negative or neutral with regard to a certain entity of interest. We define an entity as the non-abstract subject matter of a conversation or discussion, e.g., a movie or a new car model, towards which the writer can express his or her views. The classification models can be configured in a cascaded pipeline. We have to deal with several problems, such as the noisy character of the input texts, the attribution of the sentiment to a particular entity, and the small size of the training set. In addition, we study problems of the portability of the learned models across domains and languages.

This article is organized as follows. We start with a section on related research, which allows us to more sharply define our problem definition and research focus. Then, we discuss our methodology including feature selection, particularities of each language considered, machine learning, cascaded and aggregated learners, and finally active learning. The next sections regard our experiments. We describe our corpora, the evaluation measures used, the results, and their discussion. The two last sections respectively give ideas for future research and the main conclusions.

2 Related Research and Problem Definition

Early work by Hearst (1992) and Kessler et al. (1997) initiated research into classifying text according to sentiment or genre. Current systems identify the opinion of sentences in documents or of complete documents and classify these as positive, negative or neutral. In some cases other types of sentiment classifications are used (e.g., the emotions "happy, sad, anger, fear, disgust, surprise"), which reveal other aspects of the content (Huber et al. 2000, Turney 2002, Kamps and Marx 2002, Liu et al. 2003, Nijholt 2003, Whitelaw et al. 2005, Leshed and Kaye 2006). We can distinguish two main techniques for sentiment analysis of texts: symbolic and machine learning techniques. The symbolic approach uses manually-crafted rules and lexicons, whereas the machine learning approach uses supervised or weakly supervised learning to construct a model from a large training corpus, supervised models being here the most popular.

In a *symbolic setting*, a text is considered as a collection of words (occasionally multi-word expressions) often without considering any of the relations between the individual words (bag-of-words representation). The sentiment of a word is defined by a human expert, and the sentiment of a text is computed as the aggregation of the sentiment of its composing words (such as average or sum). Hatzivassiloglou and Wiebe (2000) indicated that adjectives are good sentiment indicators; their sentiment may, however, depend on the context, e.g., "a predictable plot" versus "predictable steering". Turney (2002) searches the texts on the World Wide Web to determine the semantic orientation of adjectives and nouns by using the "close to" operator, comparing the number of hits of the word as a close neighbor of respectively "excellent" or "poor". Kamps and Marx (2002) use WordNet to determine the orientation of a word. Words in WordNet can be considered nodes that are connected by synonymy relations, making it possible to construct a path between them. By

comparing the semantic similarity (Budanitsky and Hirst 2004), i.e., the length of the path from a word to respectively the words "good" and "bad", the semantic orientation of the word can be obtained. Techniques other than the bag-of-words approach include Mulder et al. (2004), who model the relation between sentiment and object, using certain mechanics that alter the intensity (e.g., intensification and quantification) or orientation (e.g., negation and certain verbs) of the sentiment.

Machine learning techniques for sentiment classification gain interest because of their capability to model many features and in doing so, capturing context (cf. Polanyi and Zaenen 2006), their more easy adaptability to changing input, and the possibility to measure the degree of uncertainty by which a classification is made. Supervised methods that train from examples manually classified by humans are the most popular. The most common approaches here use the single lowercased words (unigrams) as features when describing training and test examples. In opinion mining certain sentiments are expressed in two or more words, and the accurate detection of negation is important because it reverses the polarity. Pedersen (2001) showed that word n-grams are effective features for word sense disambiguation, while Dave et al. (2003) indicated that they are able to capture negation. In an alternative approach to negation (Pang et al. 2002), each word following a negation until the first punctuation receives a tag indicating negation to the learning algorithm. Other approaches select only a subset of the words (Wiebe 2000, Pang et al. 2002, Salvetti et al. 2004 and Chesley et al. 2006) often by considering solely adjectives detected with a part-of-speech (POS) recognizer. An opinion word lexicon was constructed by Riloff et al. (2003), while Hu and Liu (2004) identify words describing the features of a product (e.g., "The camera takes *incredible pictures*.") and use them in the classification. The supervised techniques are applied mostly for recognizing the sentiment of complete documents (e.g., Pang et al. 2002, Finn and Kushmerick 2003, Mullen and Collier 2004, Salvetti et al. 2004, Gamon 2004, Aue and Gamon 2005, Bai et al. 2005). An example of the classification of sentences is found in Dave et al. (2003). Pang and Lee (2004) use a step-wise approach for the classification of texts by first removing objective sentences from it (using a machine learning-backed minimal cut algorithm), and then classifying the remaining ones.

We only found few *weakly supervised learning approaches* in the literature. In such settings the manual labeling is limited. One example is the work of Hatzivassiloglou and McKeown (1997), in which a hypothesis is defined stating that two adjectives conjoined by "and" have the same orientation, while adjectives conjoined by "but" have an opposite orientation. Clustering can be used to determine the semantic orientation of many adjectives present in a corpus; the obtained clusters are then manually labeled. Wang and Wang (2007) identify in texts product properties (PPs) and their correlated opinion words (OWs) by using an iterative cross-training method, which starts from a small labeled part of the corpus. Two naïve Bayes classifiers (respectively for detecting PPs and OWs) are trained using contextual features (e.g., "presence of an OW/PP to the left/right" and "presence of an adjective to the left/right"). In each round the PPs and OWs with highest classification accuracy are labeled and used as training examples in the next rounds.

With the techniques described above, pretty good results can already be obtained, but nevertheless, there are important challenges that need to be overcome.

Opinion extraction from *noisy* Web texts (such as blogs) still poses problems to be researched. The blog texts exhibit a large number of anomalies. On a lexical level new words, contractions of existing words, and community jargon are no exception. On a syntactic level, we often cannot speak of real sentences. The TREC 2006 Blog track[1] aimed to explore the information seeking behavior in the blogosphere – which was modeled by mixing blogs on various topics with assumed spam blogs – and contained an opinion retrieval task. Opinion mining in legal blogs has been performed (Conrad and Schilder 2007), mood levels were detected in blog posts (Mishne 2005, Mishne and de Rijke 2006), six basic emotion categories were recognized in blog sentences (Aman and Szpakowicz 2008), but these studies did rarely involve the processing of real malformed language. In their experiments for predicting political affiliation, Mullen and Malouf (2006) use a spell check program to automatically replace words flagged as misspelled with their most likely correction. We are not

[1] http://trec.nist.gov/pubs/trec15/t15_proceedings.html.

aware of research on sentiment extraction from (noisy) Dutch or French texts.

Secondly, research has only recently shown interest in extracting the sentiments that are *expressed towards a certain entity of interest* (in our case products) or their attributes (e.g., the size of the Nokia mobile phone) (e.g., Mulder et al. 2004, Zhang and Zhang 2006, Kobayashi et al. 2007). In the same sentence different opinions might be expressed towards different entities, or towards the same entity, but with varying intensity. Both the TREC 2006 and 2007[2] Blog tracks included a task which involved locating blog posts that express an opinion about a given target. The 2007 Blog track also included a polarity subtask, requiring participants to determine the positive or negative orientation of the opinions found.

Thirdly, there is the *problem of manual annotation* of sufficient and representative training examples. In an active learning approach all the examples are labeled by a human, but the limited set of examples to be labeled is carefully selected by the machine. We start with a seed set of labeled examples, although the existence of such an initial seed set is not firmly needed in order to correctly apply the algorithm. At each iteration, one example (Seung et al. 1992, Lewis and Catlett 1994, Lewis and Gale 1994, Dagan and Engelson 1995, Freund et al. 1997, Liere and Tadepalli 1997, McCallum and Nigam 1998, Iyengar et al. 2000, Roy and McCallum 2001, Tong and Koller 2002, Baram et al. 2003, Saar-Tsechansky and Provost 2004, Osugi 2005) or a set of examples (Brinker 2003, Xu et al. 2003, Nguyen and Smeulders 2004) is selected, labeled by a human and added to the training set in order to retrain the classifier until a stop condition is reached. The selection of examples is not random. Often, the examples selected are those that the current classifier considers as most uncertain and thus most informative. Or, examples are selected that are representative or diverse with regard to the pool of unlabeled examples. We cite here a few approaches that are relevant with regard to our research. Seung et al. (1992) use a committee of classifiers to obtain a measure of classification (un)certainty for each example, considering uncertain examples to be the most informative. Lewis and Gale (1994) use a probabilistic classifier to obtain the same goal. As a variation of support vector machines Tong and Koller (2002) attempt to reduce the version space (the space of all possible hypotheses) as much as possible, when adding an example. Various attempts at reducing future classification errors have been put forward by Iyengar et al. (2000), Roy and McCallum (2001), and Saar-Tsechansky and Provost (2004). Combinations of active learning methods have been proposed that achieve both exploration of the feature space and fine-tuning of the classification boundary (Baram et al. 2003, Osugi 2005).

Finally, there is the issue of *cross-domain and cross-language classification*. Because of the lack of annotated training examples, it could be very useful if we can port a learned classification model to a new domain or new language. So, it is interesting to see the effect of using annotated examples of one subject domain (e.g., car opinions) for training a classifier that is used on an other domain (e.g., movies). This problem is already tackled by Finn and Kushmerick (2003) and Aue and Gamon (2005).[3] Overall, they show that sentiment analysis is a very domain-specific problem, and it is hard to create a domain independent classifier. One possible approach is to train the classifier on a domain-mixed set of data instead of training it on one specific domain (Finn and Kushmerick 2003). In addition, we assess how well we can port the opinion extraction technology across different languages, when we process English, Dutch and French texts. This multilingual aspect is completely novel in opinion mining.

To summarize, our goal is to determine sentiments towards a certain entity in Web sentences (written in English, Dutch and French) that often are of noisy nature and not well-formed. We use supervised machine learning techniques and will reduce the number of training examples by means of active learning. Moreover, we test and evaluate the portability of the learned classification models across domains and languages.

[2] http://trec.nist.gov/pubs/trec16/t16_proceedings.html.

[3] Despite the strong resemblance, our approach should not be confused with pure *transfer learning* where you learn a new classification model taking advantage of a model learned for a different, but related task. Transfer learning typically requires further labeled data for the new task (Raina et al. 2007).

3 Methodology

We consider opinion recognition as a supervised classification task. The objective of the classification regards the sentences of Web texts such as blog, newsgroup, forum and review texts, that mention a certain entity of interest (in our experiments below, a brand of a car or name of a movie title). From the sentences we extract relevant features for the sentiment classification. We use state of the art classification algorithms, but go deeper into the cascading and aggregating of the classification models where the models could be trained with different examples, with a different set of features, or by using a different classification algorithm. At the deep level of a classification cascade, we use expensive feature extraction techniques, such as sentence parsing, for the classification of difficult cases. At a certain level of processing detail, the features become language dependent. Examples are the treatment of negation and sentence parsing. Finally, we use techniques of active learning in order to reduce the workload in annotation.

A machine learning framework in an information extraction setting such as opinion mining offers several advantages, including a lesser building effort compared to drafting symbolic rules and a probabilistic assignment that is valuable to assess the uncertainty of the assignment and hence the need for computing expensive feature representations that rely on natural language processing. In addition, such a framework allows building a probabilistic content model of a text, which can be integrated in a probabilistic retrieval model such as a language model (Croft and Lafferty 2003).

3.1 Feature Selection

In a preprocessing phase, the texts are tokenized and segmented into sentences. Errors may occur here as the texts are not always lexically and syntactically well formed. In the sentences we recognize the entity of interest. We do not resolve noun phrase coreferents (i.e., pronouns and synonymous terms), which could improve the assignment of the sentiment within one sentence, and would allow us to find sentiments towards the entity across different sentences that do not explicitly mention the entity. Each sentence is represented by a feature vector which contains binary features (occurrence or non-occurrence of a feature) or numerical features that represent the occurrence frequency or weight of a feature. The features used in our tests include:[4]

- **Unigrams**: These are simply the words or tokens that make up the vocabulary of the sentences in our collection. We use a short multilingual stopword list of 147 words from www.publist.com, which we filtered and extended for the languages that we consider.
- **Stems**: A stem is the base part of a word, obtained by removing common morphological and inflexional endings from that word. We used Snowball[5] as stemmer, which implements Porter's algorithm (Porter 1980).
- **Negation**: Trivial solutions for negation detection are using n-grams (Dave et al. 2003) (e.g., "not worth") or tagging each word after the negation until the first punctuation (with for example "NOT_") (cf. Pang et al. 2002), the latter being implemented for our purposes. For Dutch and French, more elaborate rules were drafted including rules to limit, extend or neutralize (e.g., double negations) the negation and the handling of comparisons (e.g., in *X is better than Y* "better" would become "NOT_better" if Y is the entity of interest). Because a negation can also apply to the words before it, for Dutch and French we use a window of words that limits the maximum number of words in the context to be negated.
- **Discourse features**: Often sentences will contain multiple sentiments of which one is clearly most important, which is signaled by discourse-based connectors (Polanyi and Zaenen 2006). For instance, the part of the sentence that comes immediately after "même si" (even if)

[4] We also considered other features such as n-grams of characters; bigrams and trigrams of words (cf. Chambers et al. 2006); lemmas; punctuation patterns; word pairs that are not necessarily subsequent; words belonging to a certain part-of-speech (POS) (cf. Wiebe 2000); and the number of verbs in a sentence where a large number of verbs could signal epistemic modality and thus the possibility of a sentiment (Rubin et al. 2006). Tests showed that these features had little value when classifying the sentiment of our data.

[5] http://snowball.tartarus.org/.

will often be the least important feeling, e.g., *même si le film a eu beaucoup de succès, je le trouvais vraiment nul!* (even though the movie had a lot of success, I really found it nothing!). The most common expression is "but", but many other words are also used (e.g., "despite", "although"). A list of such expressions (of sufficient strength) is manually constructed for each language, and for each expression it is recorded whether the part of the sentence that is either following or preceding the expression is most important. All words in the sentence that are determined as being non-important are then removed. The list sizes are 4 for English, 7 for Dutch and 8 for French.

In certain cases we need additional processing of the sentence, for instance, when several (possibly different) sentiments are expressed in it and are attributed to different entities. The parse tree gives additional information on the entity for which a certain feeling is expressed. So, unigram feature weights are computed based on the position of the unigram in the parse tree relative to the entity of interest.

- **Depth difference:** The difference in depth between the word feature and the entity of interest in the parse tree determines the feature weight, which is inversely proportional with this difference. This method is used for English and Dutch.
- **Path distance:** The parse tree is seen as a graph; the weight of a word feature is inversely proportional to the length of the path between the feature and the entity of interest using a breadth-first search. This method was used for French because of the graph output format of the French parser.
- **Simple distance:** For comparison with parse tree features we define the weight of a word feature inversely proportional to its distance to the entity of interest in the sentence. This method was used for all languages.

3.2 Particularities per Language

In this section we give some particularities per language, mostly observed by annotating the Dutch and French corpora and intermediate error analyses in both languages. Based on the number of distinct words, the French language has the richest vocabulary, while in general we found that the English language was more simple in terms of vocabulary and syntactic constructions (see also Section 6.5).

Feature selection for the Dutch language requires extra knowledge rules with regard to *compound words and composed verbs*. Unlike English, in Dutch you can glue different words together freely to form a concept derived from the composing words. A relevant example would be "topfilm", translating to "top movie" in English. While it is practical that the concept is captured in a single token, it also means that more examples are needed for the classifier to learn it. As an illustration consider combinations such as "top actress", "top car" and others. A list of key sentiment indicators (e.g., "top", "super", "rot" (rotten)) commonly found in compounds was manually constructed from the corpus, and each token in the input sentence matched against it. If the token starts with a term from the list, that term (i.e., a substring of the token) is added as an extra feature. In Dutch some verbs can be split into two parts in some forms, while in other forms they appear as a single word. The verb loses its meaning if you don't have both parts (which are written as one word in the infinitive form). Common examples (infinitive, third person singular) are "tegenvallen, valt tegen" (to be below expectations) and "meevallen, valt mee" (turn out better than expected). Both parts of the verb are sought in a sentence. When found, the infinitive form of the verb is added as a feature, and the split forms are ignored.

The processing of French texts needs additional knowledge with regard to the *detection of the key part of a sentence* and the handling of the *abbreviated and phonetic language*. First of all, the variety of French expressions that contrast information in a sentence (used as discourse features) is much vaster than the variety of these expressions in Dutch and English (e.g., "mis à part", "hormis", "toutefois", "malgré", "cependant", "pas moins"...). Secondly, whether the French negation is used correctly depends mostly on the type of texts considered. It was noticeable that

writers of reviews usually use the correct and complete construction (e.g., "ne... pas/plus/jamais"), whereas people who write blogs often forget the "ne" (making sentences such as *je suis pas d'accord* (I don't agree), which is closer to spoken French). This absence of correct and structured negations means that the above negation rules are bound to make errors. Then, there is the large amount of abbreviations in French sentences (the "dirtier" the text, the more abbreviations the writer uses) (e.g., "ac" for "avec" (with) and "tjs" for "toujours" (always)). Also, words are often spelled out phonetically, again shortening them considerably and making writing faster (e.g., *L é tro bel cet voitur (Elle est trop belle cette voiture)* (She is too beautiful, this car)).

Apart from those points, there are notable differences in the way French, English and Dutch writers express their emotions about something. French and English writers often use explicit feelings such as "j'aime" (I love) or "je déteste" (I hate) while Dutch writers prefer giving adjectives to something to qualify it as good or bad. For instance, the French easily say *j'aime ce film* (I love this movie) as *ce film est bien* (this film is good) whereas in Dutch *de film is goed* (the movie is good) will be used instead of *ik hou van de film* (I love the movie). We refer the reader to the micro-level error analysis (Section 6.6) for additional differences.

We also considered feature selection techniques that based on a training set with classified examples find the features that are most discriminative for the different classes. Preliminary tests with a chi-square feature selection cut down on the number of features used and improved precision, while seriously hurting recall, so we did not further pursue this path.

3.3 Machine Learning Techniques

In the following section we discuss the supervised classification algorithms that we used in order to recognize sentiments in texts. The first two machine learners use feature vectors with numerical values, the last classifier only considers binary feature values.

A *Support Vector Machine* (SVM) (Cristianini and Shawe-Taylor 2000) operates by constructing a hyperplane with maximal Euclidean distance to the closest training examples. This can be seen as the distance between the separating hyperplane and two parallel hyperplanes at each side, representing the boundary of the examples of one class in the feature space. It is assumed that the best generalization of the classifier is obtained when this distance is maximal. If the data is not separable, a hyperplane will be chosen that splits the data with the least error possible. An SVM is known to be robust in the event of many (possibly noisy) features without being doomed by the curse of dimensionality and has yielded high accuracies in sentiment classification (e.g., Pang et al. 2002). The Weka[6] implementation was used.

A *Multinomial Naïve Bayes* (MNB) classifier (Manning et al. 2008) uses Bayes rule as its main equation, under the naïve assumption of conditional independence: each individual feature is assumed to be an indication of the assigned class, independent of each other. An MNB classifier constructs a model by fitting a distribution of the number of occurrences of each feature for all the documents. We selected this classifier because of its simple implementation, its computational efficiency and a straightforward incremental learning. We used the MNB classifier of Weka.

A *Maximum Entropy* (ME) classifier (Berger et al. 1996) tries to preserve as much uncertainty as possible. The models that fit the training examples are computed, where each feature corresponds to a constraint on the model. The model with the maximum entropy over all models that satisfy these constraints is selected for classification. We choose to work with an ME classifier because in information extraction from natural language texts, this classification algorithm often yields very good results, as it can deal with incomplete information in a very elegant way. When classifying natural language utterances the training examples seldom cover all variant linguistic expressions that signal certain semantics. The Maxent[7] package from OpenNLP was used as implementation.

[6] http://www.cs.waikato.ac.nz/∼ml/weka/.
[7] http://maxent.sourceforge.net/.

3.4 Cascaded and Aggregated Learners

We can learn several classification models from the data. Especially complex classification tasks might benefit from being split in different subtasks and by using different models. We can combine the classification models in a cascade or pipeline framework, or in an aggregated fashion. The models can differ by a variety of parameters, to meet specific classification needs. The parameters mainly regard the classes in which is being classified, the example set used for training, the features (including processing and selection) by which training and testing examples are described, the type of classification algorithms and possibly additional heuristics.

The classifiers can be combined in a *pipelined cascaded way*, i.e., the output of one classifier determines whether an example will be an input to the next classifier, or whether the classification is final (cf. Viola and Jones 2001). For instance, the degree of certainty by which an example is classified might be a decision criterion. A cascade can be beneficial in various ways; most notably with regard to efficiency. A cascade may prevent heavy computations from being executed for each example in the first layer of the cascade, especially when they are not needed for the case at hand. Using a cascade, certain conditions (e.g., the certainty of the previous classification) can guard against going deeper into the cascade, where complexity rises.

We cannot only combine classification models on a vertical axis, on a horizontal axis the results of several different classification *models can be aggregated* based on certain constraints or rules. For instance, only when two or more classifiers recognize the example unanimously above a threshold certainty in a certain class, the classification is accepted. Aggregation and combination rules of different classification models might be learned from a training set. In such cases, we often speak of a meta-learner. In the experiments below we did not use a meta-learner. In case different example sets are used to train the classifiers the results of which are aggregated, we speak of a "bagged" approach.

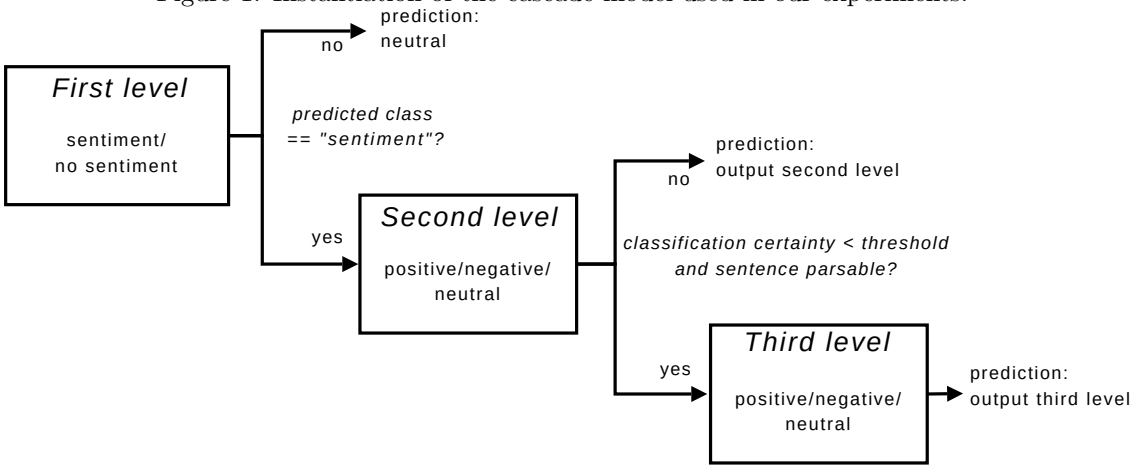
3.4.1 Experimental Cascade Architecture

We describe here the cascade architecture of our system. In the first layer of the example cascade we tackle the problem of having an abundance of neutral examples, of which a large portion are uninformative or advertisement messages. An "uninformative" example may, for instance, contain part of a website menu, or it may just be a collection of words and symbols without structure, and usually does not contain any sentiment. Another category are advertisements: i.e., snippets of text that try to convince the reader to take some action related to the entity of interest (e.g., to buy a car from a certain dealer). The first layer consists either of one classifier or of a bagged aggregation of several classifiers, the latter with the purpose of obtaining multiple proofs that an example is actually neutral. In the second layer of the cascade, our three-class classification into positive, negative and neutral sentiment is performed. In the third step of the layer, classification of difficult cases is based on the extraction of expensive features obtained by parsing the sentence into its syntactic dependency structure. A graphical representation of this setup can be seen in Fig. 1.

A sentence of the test corpus will then be processed by the cascade. In the first layer, when an example is classified as neutral by the sole classifier or by all three bagged classifiers (in the alternative setting), the classification of this sentence as neutral will be final. Otherwise, the sentence progresses down the cascade to the second layer. If the sentence is classified here with a certainty level above or equal to a preset threshold (determined empirically on a small validation set), the classification of the sentence is final (see below how the certainty of the classification is computed). In the other case the sentence is parsed. If the parse fails, the classification of this sentence by the second layer is kept. Such a sentence often concerns an uninformative sentence that survived the filter in the first layer. If the parse succeeds, the sentence percolates down the third layer and its classification is finalized. In the tests, the likelihood of each sentiment class is computed for a sentence, and the class with maximum certainty is selected.

All classifiers involved in this architecture are trained on the complete training set, or alternatively in case of second or third layer classifiers on the training examples that arrive in the

Figure 1: Instantiation of the cascade model used in our experiments.



respective layer. Note that in the latter case, we learn from representative training examples, but their number is restricted. When opting for aggregating three classifiers in the first layer, we use all neutral examples available and divide them in three bags each of them also containing all positive and negative training examples.

The above only constitutes an example of a possible architecture for classifying sentiments expressed in blogs and other Web data, which we have used in the experiments below. In these experiments we have also tested and cross-validated parts of this architecture in order to assess the usefulness of the different layers. Eventually, the selection can be reduced to one layer and to one classifier using a particular feature set.

3.5 Active Learning

The bottleneck of our supervised approach to sentiment classification is the human effort needed to annotate sufficient training examples for each language or domain. The number of annotations required to accurately predict the class label increases with the amount of different patterns present in the language used. Especially texts having poor spelling, neglecting grammatical rules and employing a community jargon, are expected to contain a large variety of patterns.

The aim of *active learning* (AL) is to obtain good results for supervised classification methods, while reducing the amount of examples to label to a minimum. So, in contrast to completely unsupervised methods (which do not require any training examples), there is still human input, but the examples to annotate are automatically selected from the set of unlabeled examples. This selection is performed in an iterative way until some stopping criterion is fulfilled (cf. Zhu et al. 2008). Almost all AL methods start their selection of examples from scratch, taking but a handful of labeled instances as seed set. While it is possible (and sometimes evaluated in the literature) to start from a larger set, the gains of AL are typically largest in this early phase of building the corpus. Good overviews of AL are found in Dagan and Engelson (1995) and Baram et al. (2003). In the following section we describe Uncertainty Sampling, Relevance Sampling and Kernel Farthest First, the AL techniques that seemed valuable to us in the frame of our goals.

3.5.1 Uncertainty Sampling

Uncertainty sampling (US) (Lewis and Gale 1994) in general regards the selection of an unlabeled example for labeling of which the currently trained classifier is most uncertain, when predicting the label. By annotating this example and adding it to the training set, it is hoped that the next classification model better classifies similar examples.

The measure of uncertainty depends on the method used. The easiest way to acquire it is by

simply using a probabilistic classifier (MNB and ME classifier in our case) that not only classifies the examples, but provides a certainty of class membership as well. For an SVM, uncertainty can be measured as the distance of an example to the separating hyperplane (Tong and Koller 2002). As an SVM performs classification in two classes, we train one SVM for each possible combination of classes (e.g., pos-neg, pos-neu, neg-neu). The examples with lowest minimum distance over all binary classification outcomes, will be selected for labeling. These are the examples that might discriminate best between two classes.

Following this method, we add informative examples to our training set. By not selecting examples for which the classifier is already certain, *redundancy in the corpus* (i.e., multiple examples following similar patterns) will be reduced. Another reason for an uncertain classification is the ambiguity of the example (e.g., a sentence expressing both a positive and negative sentiment towards the entity of interest, or a sentence expressing conflicting sentiments towards two entities, one of which is the entity chosen). These examples are interesting when more advanced parse features are used.

3.5.2 Relevance Sampling

Relevance sampling regards the *labeling of those examples which are most likely to be class members* (Lewis and Gale 1994). We use this technique to acquire more examples from a class of which we lack sufficient training data. We use a probabilistic classifier (MNB) to select the examples classified with the highest certainty, and put them into the new training set after manual labeling. The technique is similar to the relevance feedback approach in information retrieval (Huang et al. 2008).

3.5.3 Kernel Farthest First

Next to using uncertainty values, geometrical distances in the feature space give us valuable information. In the *Kernel Farthest First* (KFF) method (Baram et al. 2003), the example that is farthest from the set of examples already labeled, is picked for labeling. "Farthest" is defined as the maximum of the minimum distances of the example to each example in the set of labeled examples. The distance between two examples is given by:

$$\sqrt{K(x,x) + K(y,y) - 2 \times K(x,y)} \tag{1}$$

where $K(x, x')$ is a kernel function. In the case of a simple linear kernel, the function is the dot product of x and x'. Non-linear kernels (e.g., polynomial and radial basis functions) have the ability to map the vectors into a higher dimensional feature space. We define *Kernel Average Farthest First* (KAFF) as an alternative. In this method the distance is determined by the average of the distances to each example in the labeled set.

Selecting diverse examples is advantageous for a variety of reasons. Diverse examples are often very informative. Their identification is especially useful when the training set is not sufficiently large to predict reliable uncertainty estimates (e.g., when bootstrapping from a very small seed set). Additionally, newly-appearing types of expressions (e.g., Internet slang, buzz words) can be detected, if they are different from the expressions commonly used before. Effectively finding these patterns is useful for keeping the classification models up to date.

In an alternative setting we define a *Kernel Closest First* (KCF) metric, which is the opposite of KFF, in order to select more examples close to those of a specific class (as was the case for relevance sampling).

4 Description of the Corpora

We collected three corpora – respectively composed of English, Dutch and French sentences – from blog, review and news forum sites. There is no clear border between the types of text they represent; all sources may contain reader comments and news forum sites often contain

reviews as well. Sources include major blog sites, e.g., skyrock.com, livejournal.com, xanga.com, blogspot.com; review sites, e.g., amazon.fr, ciao.fr, kieskeurig.nl; and news forum sites, e.g., fok.nl, forums.automotive.com.[8] The sources reflect realistic settings for sentiment extraction. They contain a mixture of clean and noisy texts. The noisiness is not necessarily linked to the type of source, but rather to the presence of advertisements or to the particular style of an author. As entities we considered car brands and movie titles. These entity types are about equally distributed in the selected corpora. Each sentence was annotated and the annotation was controlled by a second person (inter-annotator agreement measured with $\kappa = 82\%$). Disagreement was resolved through mutual consent. For more efficiently constructing training and test sets for our classification, we replace an individual name by its general class (e.g., all car brands are replaced by the class "car"). Sentences with different sentiments towards different entities (e.g., film_X to be much more enjoyable than film_Y) are added twice to the training corpus, once annotated towards film_X (positive) and once towards film_Y (negative). It is also the case that different sentiments are presented towards the same entity. In case one sentiment clearly has the upper hand, the sentence was classified in this category. The annotators very rarely (in the English corpus the occurrence was 0.2%) encountered a sentence in which both a positive and negative sentiment were expressed towards the same entity of interest and where both feelings were about equally strong. These sentences were ignored. In addition, when collecting and annotating the corpus some obvious redundant examples were removed. We also removed question sentences (e.g., *What things have gone wrong with the car?*), because they do not explicitly state a sentiment. For each language we collected 750 sentences that express a positive feeling towards the entity and 750 sentences that express a negative one. Neutral examples are available in abundance in our sources; in our experiments we use 2500 instances.

When text is automatically gathered from the World Wide Web, one can expect a fair amount of uninformative text (e.g., advertisements, web site menus, links, ...). This uninformative text may be mixed with information we are interested in, making it more difficult to filter it out. Also, the language used by the writers is sometimes of poor quality, containing lots of Internet slang and misspellings, or phonetic spelling as is shown in the following examples:

(English) *The movie really seems to be spilling the beans on a lot of stuff we didnt think we hand if this is their warm up, what is going to get us frothing in December*
(Dutch) *de grote merken mogen er dan patserig uitzien en massa's pk hebben maar als de bomen wat dicht bij elkaar staan en de paadjes steil en bochtig,dan verkies ik mijn Jimny.*
(French) *L é tro bel cet voitur Voici tt ce ki me pasione ds ma petite vi!!!é tt mé pote é pl1 dotre truk!!!Avou de Dcouvrir*

5 Evaluation

As a first evaluation measure we simply take the classification *accuracy*, meaning the percentage of examples classified correctly. We also computed *precision* and *recall* of the identification of the individual classes (here positive, negative and neutral). $D_{l_{\mathrm{X}}}$ stands for "documents labeled by the human as X", $D_{c_{\mathrm{X}}}$ for "documents classified by the machine as X":

$$Precision_{\mathrm{X}} = \frac{|D_{l_{\mathrm{X}}} \cap D_{c_{\mathrm{X}}}|}{|D_{c_{\mathrm{X}}}|} \quad Recall_{\mathrm{X}} = \frac{|D_{l_{\mathrm{X}}} \cap D_{c_{\mathrm{X}}}|}{|D_{l_{\mathrm{X}}}|} \tag{2}$$

Recall and precision values can be combined in an F-measure:

$$F_{\alpha} = \frac{(1+\alpha) \times \ precision \times \ recall}{\alpha \times \ precision + \ recall} \tag{3}$$

[8] No standard annotated corpora exist for entity based sentiment analysis (including the neutral class) on sentences. Our corpora are currently proprietary: we have asked the company Attentio to publicly release the data.

Table 1: Results in terms of accuracy on the movie review corpus for different machine learning methods using a selection of features. For the bigram of words feature only those occurring at least four times were included in the feature vector.

Features	**SVM**	**MNB**	**ME**
Unigrams	85.45%	81.45%	84.80%
Unigrams & BSubjectivity	**86.35%**	83.95%	**87.40%**
Bigrams	85.35%	83.15%	85.40%
Adjectives	75.85%	82.00%	80.30%

where $\alpha = 1$ giving an equal weighting to recall and precision (F_1). Both recall and precision of positive and negative feelings are important when collecting accurate statistics on the expressed sentiments. In addition, for applications that quote the sentiment towards certain people or products, the precision of the classification of the quotes is pertinent.

In our experiments below, the cascade, its components and single classifiers are tested using 10-fold cross-validation of our complete data set for each language. At each of the 10 testing phases, 10% of the annotated corpus is used as test set, while 90% is used for training. Each test set consists of examples not selected before, leading to the full corpus (see Section 4) being tested in the process.

6 Classification Results

In order to assess the validity of our feature extraction methods, we first performed some tests on a standard corpus. We then conducted a large number of experiments on the Web corpus described in Section 4, followed by a micro-level error analysis of the classification of the sentences as positive, negative or neutral towards the entity of interest. Finally, we provide an overview of separate experiments on active learning.

6.1 Preliminary tests with a standard corpus

These first tests were performed on the Movie Review Data corpus[9] (cf. Pang and Lee 2004). The corpus contains 2000 movie reviews, that are characterized by a varied vocabulary with a typical mix of words that describe the storyline and words that describe the evaluation of that particular movie. In contrast to our task, where we classify individual sentences as positive, negative or neutral, here entire reviews have been classified by the original dataset preparers according to a positive and negative sentiment. The results are given in Table 1 (up to 87.40% accuracy), where also the types of features that we used in these tests are indicated. Here "BSubjectivity" stands for a simple subjectivity analysis using an MNB classifier – trained on the subjectivity dataset also introduced by Pang and Lee – which removes objective sentences from the reviews prior to their classification.

We see that the ME and SVM classifiers combined with subjectivity analysis perform best overall (the difference in their performance was not statistically significant[10]). Improvements resulting from subjectivity analysis had a confidence level $\geq 90\%$ for SVM and $\geq 99.5\%$ for MNB and ME.

The results in Table 1 are comparable with the ones found in the literature that use this corpus. Ling-Pipe[11] reports an accuracy of 81.5% using a polarity (positive/negative) classifier on character 8-grams, and 85% when combining it with a subjectivity analyzer. Both results are obtained *without cross-validation*; instead they put every tenth review in the test set. According to Pang and Lee, the best performance (using subjectivity and polarity) reported is 86.4% in terms

[9] Available at http://www.cs.cornell.edu/People/pabo/movie-review-data/.
[10] All confidence level tests are obtained with two-paired t-test.
[11] http://www.alias-i.com/lingpipe/demos/tutorial/sentiment/read-me.html.

of accuracy. The reasons we perform slightly better are likely due to different implementations and/or feature settings. Also, Pang and Lee don't report any tests using subjectivity analysis in combination with an ME classifier.

6.2 General results of tests with our corpora

We performed a large number of experiments in order to most confidently answer the set of research questions mentioned below.

Table 2 gives the best performances for each language. We compare a cascaded approach with three single classifiers (SC). In the cascaded approach unigram features are used in the first layer, unigram, discourse and negation features are used in the second and third layers. In the third layer parse features are added. We give the results of the classification with the complete cascade, i.e., *Cascade with layers 1, 2 and 3*, and results of using the cascade with the two first layers, i.e., *Cascade with layers 1 and 2*. The effect of layer 3 in combination with layer 2 is separately tested (see below). *SC uni-lang* is identical with the classifier used in layer two of the cascade in terms of features, training examples and classification algorithms used. *SC uni-lang-dist* simulates properties of layers 2 and 3 of the cascade by taking into account the distance of a term from the entity of interest in terms of number of words (simple distance feature). *SC uni* uses a simple unigram-based classifier. Removal of stopwords and of the entity of interest is always performed when constructing the feature vector.

We experimented with the three classification algorithms mentioned above and found the best results when using an MNB classifier for English, an SVM (linear kernel)[12] for Dutch and an ME classifier for French[13]. The first and second layer of the cascade and the single classifiers were trained with nine tenth of the corpus. The third layer of the cascade was trained with all parsable positive and negative examples that were classified correctly in an isolated cross-validation of the second layer. When testing the cascade, examples with a certainty above a given threshold move to layer three.[14] Examples that cannot be parsed in the third layer of the cascade keep the classification given by the second layer.

The settings of the tests mentioned below are the same as the ones described here unless otherwise stated.

6.3 Benefits and disadvantages of the cascaded architecture

Question 6.3.1. *Is using a cascaded architecture of classifiers valuable and in what circumstances?*

Table 2 shows already good results based on single classification models (rows 3-5), although with careful selection of the features. The reason for using a more complex classification architecture is that we can train classifiers that focus on a particular problem (e.g., the attribution of a sentiment to an entity in the third layer of our cascade). However, we should have enough representative training examples in order to learn an accurate model that really makes the difference. This is demonstrated by the only very small improvement (compare row 1 to row 3). For English and French differences are detected with confidence level $\geq 99.95\%$ except for the recall of English positive and negative examples.

To completely answer the above question we split it in several sub-questions.

Question 6.3.2. *Does it help to build a first filter for neutral sentences?*

Table 3 shows the effect of the cascade in the filtering of neutral examples.[15] The pipelined approach of two layers especially improves the recall (confidence level $\geq 99.95\%$) of the neutral at the expense of a reduced precision (compare row 2 to row 3). The F-measure overall improves

[12] We used an error tolerance of 0.05 for all experiments.

[13] Note the ME classifier does not allow parse feature weights within the binary feature functions.

[14] We tested different uncertainty values on a validation set that threshold the percolation to the third layer, yielding a threshold of 75% for the MNB classifier used for English, 33.4% for the ME classifier used for French, and a hyperplane distance of 0.11 for the SVM classifier used for Dutch.

[15] Note that no bagging (see Question 6.3) was used in these experiments.

Table 2: Our best results in terms of accuracy, precision, recall and F-measure (F_1) using the English (a), Dutch (b) and French (c) corpora. For English, Dutch and French we implemented respectively an MNB, an SVM and an ME classifier – 10 fold cross-validation.

(a) English

Architecture	**Accuracy**	**Precision** pos/neg/neu	**Recall** pos/neg/neu	**F-measure** pos/neg/neu
Cascade with layers 1, 2 and 3	83.30	69.09/85.48/85.93	55.73/82.40/91.84	61.70/83.91/88.79
Cascade with layers 1 and 2	83.10	70.49/87.72/84.61	54.13/79.07/93.00	61.24/83.17/88.61
SC uni-lang	83.03	69.59/86.77/85.08	56.13/79.60/92.12	62.14/83.03/88.46
SC uni-lang-dist	80.23	60.59/78.78/86.57	59.87/82.67/85.60	60.23/80.68/86.08
SC uni	82.73	68.01/85.63/85.53	58.40/78.67/91.24	62.84/82.00/88.29

(b) Dutch

Architecture	**Accuracy**	**Precision** pos/neg/neu	**Recall** pos/neg/neu	**F-measure** pos/neg/neu
Cascade with layers 1,2 and 3	69.03	63.51/53.30/72.20	42.93/31.20/88.20	51.23/39.36/79.40
Cascade with layers 1 and 2	69.80	66.60/58.31/71.66	41.73/29.47/90.32	51.31/39.15/79.92
SC uni-lang	69.05	60.39/52.59/73.63	49.60/33.87/85.44	54.47/41.20/79.10
SC uni-lang-dist	68.85	61.08/54.52/72.20	43.73/30.53/87.88	50.97/39.15/79.27
SC uni	68.18	58.73/49.58/73.24	48.00/31.73/85.16	52.82/38.70/78.75

(c) French

Architecture	**Accuracy**	**Precision** pos/neg/neu	**Recall** pos/neg/neu	**F-measure** pos/neg/neu
Cascade with layers 1, 2 and 3	67.68	50.74/55.88/71.90	27.47/38.67/88.44	35.64/45.71/79.32
Cascade with layers 1 and 2	67.47	52.69/53.96/71.56	26.13/38.13/88.68	34.94/44.69/79.21
SC uni-lang	65.97	47.67/50.33/72.18	30.00/40.67/84.36	36.82/44.99/77.79
SC uni-lang-dist	65.97	47.67/50.33/72.18	30.00/40.67/84.36	36.82/44.99/77.79
SC uni	65.83	45.67/50.82/72.23	28.80/41.33/84.28	35.32/45.59/77.79

slightly. More specifically, 19 more non-neutral examples were wrongly classified as neutral (e.g., *A Ferrari is not cheap to buy or run and residual values weaken if you use the car regularly.*), while 22 more neutral examples got correctly recalled (e.g., *BMW 740i car horn compact design by Piaa*). The results do not completely correspond with the intended goals of the first filter, i.e., having high recall of positive and negative examples, and high precision of the neutral examples that are finally classified in this layer.

Question 6.3.3. *What is the effect of the bagging?*

A bagged aggregation of the first layer (Section 3.4.1) where we trained three classifiers with different bags of 3500 neutral examples and where examples on which all three aggregated classifiers agree of being "neutral" are definitively classified under this category, did not improve the overall results (bagging is not used when obtaining the best results in Table 2). However, Table 4 shows that for English the bagging helps when one wants to filter neutrals with high precision, still achieving good recall for positives and negatives (confidence level $\geq 99.95\%$).

Question 6.3.4. *Does the cascade help with efficiency?*

Table 3: Results with regard to the classification of neutral sentences using the English (a), Dutch (b) and French (c) corpora – 10 fold cross-validation.

(a) English

Architecture	**Precision**	**Recall**	**F-measure**
Layer 1 of the cascade	88.79	86.20	87.48
Layer 1 and 2 of the cascade	84.61	93.00	88.61
Layer 2 of the cascade	85.08	92.12	88.46

(b) Dutch

Architecture	**Precision**	**Recall**	**F-measure**
Layer 1 of the cascade	74.49	82.00	78.07
Layer 1 and 2 of the cascade	71.66	90.32	79.92
Layer 2 of the cascade	73.73	85.88	79.34

(c) French

Architecture	**Precision**	**Recall**	**F-measure**
Layer 1 of the cascade	75.95	81.36	78.56
Layer 1 and 2 of the cascade	71.56	88.68	79.21
Layer 2 of the cascade	72.18	84.36	77.79

Table 4: Results of the the first layer (English corpus) – 10-fold cross-validation.

Features	**Precision neu/not neu**	**Recall neu/not neu**	**F-measure neu/not neu**
Using bagging	96.05/51.69	62.15/94.06	75.47/66.71
No bagging	88.79/78.07	86.20/81.87	87.48/79.92

Reserving processing that is computationally expensive to a subset of sentences reduces the computational complexity. If meanwhile also the performance of the classification is improved, we can state that the cascade has a positive effect on performance. We can see in Table 2 that for English and French both the first and third layers provide an improvement over a one-layered approach (compare rows 1-2 to rows 3-5). For the 4000 English/French sentences in the first layer (least expensive features), 2427/2678 sentences receive their final (neutral) classification. The complexity of the feature extraction in the second layer increases linearly due to additional modules iterating over the examples' word tokens, while the increase in number of features is negligible. Consider however alternative setups where POS-tagging is performed or bigrams are extracted, significantly increasing complexity and the number of features respectively. Only 245/306 English/French examples were passed on to the expensive third layer. As an illustration, the efficiency bottleneck for English is in the third layer's parsing, which takes about 7 seconds on average for each sentence, when parsing is performed on an Intel Core2 Duo E6400 processor with 2GB of RAM.

6.4 Portability across domains

Question 6.4.1. *Does a domain-specific training improve the results and can we port a model trained on one domain to another domain?*

All experiments were conducted on a mixture of the car and movie domains. The meaning of certain words might be domain specific and might infer different sentiments in different domains. In Table 5 tests performed on both domains separately are shown for Dutch. Corpus distribution are 450 positive, 450 negative and 1250 neutral examples for each domain. For our texts, in contrast to findings of the literature (Aue and Gamon 2005) the results degrade when a separate domain is considered, which can be explained by the fewer training examples used. Table 5 also includes results of the classification of sentences from the car domain using a cascade model trained

on the movie domain and vice versa. We see that using a classification model trained on the car domain gives the best results overall. For example, the results of classifying the sentences from the movie domain using this model are clearly better than those using a model trained on the movie domain (compare line 4 to line 1 in Table 5(b)). We assume this is because in the car domain, more accurate patterns are learned. The movie domain yields ambiguous features, as sentences often contain a mixture of positive, negative and objective information (Pang et al. 2002).

Table 5: Results in terms of accuracy, precision, recall and F-measure (F_1) on the car (a) and movie (b) domain corpora for Dutch. SVMs were used – 10 fold cross-validation.

(a) Car domain

Architecture	**Acc**	**Precision** pos/neg/neu	**Recall** pos/neg/neu	**F-measure** pos/neg/neu
Cascade with layers 1, 2 and 3	70.65	74.04/62.32/71.45	55.78/39.33/87.28	63.62/48.23/78.57
SC uni-lang	70.84	69.67/63.02/72.83	60.22/43.56/84.48	64.60/51.51/78.22
SC uni-lang-dist	70.51	70.23/65.06/71.53	54.00/38.89/87.84	61.06/48.68/78.85
Cascade layers 1, 2 and 3 trained on movie domain	63.95	62.33/48.47/65.72	40.44/17.56/89.12	49.06/25.77/75.65

(b) Movie domain

Architecture	**Acc**	**Precision** pos/neg/neu	**Recall** pos/neg/neu	**F-measure** pos/neg/neu
Cascade with layers 1, 2 and 3	56.88	46.15/31.76/59.85	12.00/12.00/89.2	19.05/17.42/71.64
SC uni-lang	59.77	44.76/48.09/65.29	31.33/33.56/79.44	36.86/39.53/71.67
SC uni-lang-dist	62.05	48.70/51.81/66.04	29.11/31.78/84.80	36.44/39.39/74.26
Cascade layers 1, 2 and 3 trained on car domain	59.40	55.61/36.18/63.98	25.33/23.56/84.56	34.81/28.53/72.85

6.5 Linguistic processing and portability across languages

Question 6.5.1. *Does it help to build a classifier with expensive sentence parse features?*

Parse features – given that they can be correctly identified – should more accurately link sentiments to entities. Because parsing fails on quite a substantial amount of sentences, we can only consider parse features in a cascaded approach. For the tagging of the grammatical category of a word and the parsing of individual sentences into its dependent constituents, we used Charniak[16] parser for English (which failed on about 1/5 of our sentences), Alpino[17] for Dutch (the failure rate was here about 1/3) and Syntex[18] for French (with a failure rate of 1/20).

We separately performed tests with ambiguous sentences (i.e., sentences with conflicting sentiments towards different entities) of the Dutch corpus that could be parsed (102 out of the 182 Dutch ambiguous sentences). We assume that the high failure rate is due to the noisy character of the texts. Table 6 demonstrates the difficulty of correctly classifying ambiguous sentences, but shows a small improvement with regard to F-measures when the parse features for classifying these sentences were incorporated (confidence level $\geq 90\%$ for pos, $\geq 95\%$ for neg/neu), notwithstanding the parsing errors. Note that despite the positive effect on the classification of ambiguous sentences, the overall results for Dutch sentences did not improve by using parse features in layer three. We may not forget here that the training set has a substantial influence in the classification. The parse features used do in fact still provide some benefit over the features used in layer two

[16] http://www.cs.brown.edu/people/ec/.
[17] http://www.let.rug.nl/ kleiweg/alpino/index1.html.
[18] http://w3.univ-tlse2.fr/erss/textes/pagespersos/bourigault/syntex.html.

for these sentences, but the training set in layer three consisting of examples where parse features could play a role, is simply too small (about 250 examples on average for all classes combined) to produce an accurate classifier. Another factor is the high variance in the depths of the sentence's tokens in the parse tree in Dutch sentences; more fine grained features based on sentence compression techniques (Knight and Marcu 2000, Galley and McKeown 2007) might here help to improve results.

Table 6: Results with regard to the classification of ambiguous positive, negative and neutral sentences (Dutch corpus) that can be parsed – 10-fold cross-validation.

Features	**Precision pos/neg/neu**	**Recall pos/neg/neu**	**F-measure pos/neg/neu**
Unigrams	28.57/100/30.43	6.06/ 7.69/93.33	10.00/14.29/45.90
Unigrams + parse features	33.33/100/32.18	9.09/15.38/93.33	14.29/26.67/47.86

Table 7: Results with regard to the classification of uncertain positive, negative and neutral sentences (English corpus) that can be parsed – 10-fold cross-validation.

Architecture	**Precision pos/neg/neu**	**Recall pos/neg/neu**	**F-measure pos/neg/neu**
Cascade with layers 1, 2 and 3	50.48/69.05/57.14	71.62/86.14/11.43	59.22/76.65/19.05
Cascade with layers 1 and 2	53.95/78.48/41.11	55.41/61.39/52.86	54.67/68.89/46.25

Question 6.5.2. *Does the uncertainty level have an impact on the performance and how well does it discriminate examples for which parse features are necessary in the sentiment recognition?*

The pipeline between the second and the third layer is constrained by a threshold certainty level, i.e., the examples that are processed by the third layer were classified by the second layer with an uncertainty that is equal or higher than a preset value. We performed separate experiments with levels 2 and 3 of the cascade. For instance, for French 24% of the examples were processed by the third layer, including 37% of the ambiguous examples where a different sentiment is expressed towards different entities. Separate tests on the English corpus (Table 7) showed a significant ($\geq$ 95%) improvement in recall and F-measure of negative sentiments when complementing the cascade with a third layer. As only 2.8% of all neutral examples are processed by the last layer, the bad F-measure for neutral has but a minor effect on the total statistics.

Question 6.5.3. *How can we best classify difficult examples on which the parsing fails?*

Sentences making it past level 2 of the cascade on which the parsing fails, might still contain uninformative text or advertisements. In our architecture, non-parsable sentences keep the classification of the second layer. An alternative cascade setup that classifies these sentences by default as neutral (results not included in the table) gives worse performance (e.g., for English 70 positive, 21 negative and 78 neutral sentences would receive the neutral classification, resulting in an accuracy well under 50%).

Question 6.5.4. *What is the effect of the noise in our results?*

From Table 2 it is clear that it is more difficult to extract the correct sentiment from Dutch and French sentences. One of the reasons is certainly the rich and varied vocabulary and syntax employed in these languages reflected in proportions of respectively 15% and 14% unique words in the corpus, compared to 10% for English. We need many more training examples to compensate for the large variety in language features and to accurately identify all sentiment patterns.

The French texts especially diverge from formal language (see example in Section 4). Certain sources such as skyrock.com are notorious for only containing texts in a phonetic community language. In the tests shown in Table 8, 1250 neutral, 750 positive and 50 negative examples from our standard French corpus (see Section 4) are switched with texts from this source. We see that the results in terms of recall of positive examples are very bad, which can be explained by the large variation in language use. Upon inspection of the results it was noted that buzz words (e.g., "bien", "tro", "trop", "belle", "super") were frequently used with a consistent spelling, which improved precision of especially positive examples.

Table 8: Results with regard to the classification of very noisy sentences that diverge from formal language (French corpus) – 10-fold cross-validation.

Architecture	**Precision pos/neg/neu**	**Recall pos/neg/neu**	**F-measure pos/neg/neu**
Cascade with layers 1, 2 and 3	60.87/52.63/83.41	24.28/15.87/97.11	34.71/24.39/89.74
SC uni-lang	61.25/55.00/83.90	28.32/17.46/96.56	38.74/26.51/89.78
SC uni-lang-dist	61.25/55.00/83.90	28.32/17.46/96.56	38.74/26.51/89.78

Question 6.5.5. *What is the effect of the language specific features?*

Table 2 confirms a slight improvement of performance compared to the baseline (*SC uni*) when using language specific features (*SC uni-lang*)[19]. We performed a number of additional tests comparing the performance of a single classifier with unigram feature (i.e., *SC uni*) to the same classifier enhanced with a language-specific feature in order to assess its value (results not included in any table). Negation and stemming had an overall significant positive effect for English (confidence level $\geq 99\%$). For Dutch negation and stemming only proved beneficial for improving recall (confidence level $\geq 99.5\%$), while stemming has also a positive influence on accuracy. The same influence of stemming could be confirmed for French, but negation only positively influenced precision (confidence level $\geq 97.5\%$). Discourse features (for all languages) and Dutch-specific features such as resolution of compounds and composed verbs, only yielded a very slight improvement which was statistically not significant.

For the influence of the parse features we refer to Question 6.5.1.

6.6 Additional Error Analysis

An error analysis was performed on the results of the cross-validation in the second layer of the cascade for the English, Dutch and French corpus (cf. SC uni-lang). The first 50 misclassifications from positive into negative or vice versa, were looked at. The most likely or most apparent cause for not correctly classifying the example was noted. Table 9 gives an overview of (sometimes related) causes. We mainly cite examples from the English corpus, but the error categories also apply for Dutch and French texts. In the examples given below, important features indicating the annotated sentiment are underlined, whereas features indicating the (wrongly) predicted sentiment are in **bold**.

1. Natural language and especially the language used in informal communications is enormously varied. The most important cause of misclassification is the lack of training examples, i.e., the lack of classification patterns. As a consequence, words that should not have any sentiment connotation can in fact be oriented towards positive or negative due to an accidentally higher occurrence of these words in a positive or negative context in the training set. In addition, words that do indicate sentiment might occur rarely or not at all. The lack of training patterns also plays a role in many of the phenomena mentioned below.

[19] Confidence levels with respect to accuracy: English $\geq 80\%$, Dutch $\geq 90\%$ and French $< 75\%$.

Table 9: Error analysis based on examination of 50 misclassified sentences in English, Dutch and French.

Id	Cause	English	Dutch	French	All
1	Features insufficiently known and/or wrong feature connotations	23	21	15	59
2	Ambiguous examples	12	8	8	28
3	Sentiment towards (sub-)entity	3	3	9	15
4	Cases not handled by negation	3	3	4	10
5	Expressions spanning several words	3	5	2	10
6	Understanding of the context or world knowledge is needed	2	2	4	8
7	Domain specific	0	3	3	6
8	Language collocations	2	2	2	6
9	A sentiment feature has multiple meanings	2	1	2	5
10	Language specific	0	2	1	3

2. Ambiguous examples are examples in which positive and negative keywords are present. It may be clear to a human which is the prevailing sentiment, but the task is difficult for a machine.
   E.g., *A Good Year is a **fine** example of a **top-notch** director and actor out of their elements in a sappy romantic comedy lacking in ...*

3. Another important problem regards the sentiment not being related to the entity of interest.
   E.g., *certainly more comfortable and rewarding than an Audi Q7 and ...*

4. Negation tagging, while helpful, is not perfect, and some cases may still be missed or handled wrongly because of ignorance of the specificities of the language.

5. Sentiments can also be expressed more directly with the use of (possibly metaphorical) expressions spanning multiple words, where the full expression (but no additional context) is needed to correctly interpret the sentiment. For instance, the Dutch "dit is een schot in de roos" ("this is a shot in the bull's eye"), meaning they got it exactly right.

6. Sometimes the sentiment of a statement is not clear from the words used, but can only be inferred either from its context or from world knowledge.
   E.g., *I don t know maybe it s because I was younger back then but Casino Royale felt more like a connect the dots exercise than a Bond movie.*

7. Domain specific knowledge is needed to catch the sentiment of a word. In the given example "dormir" (to sleep) is what you do when a movie is boring.
   E.g., *[...] attention pour avoir une chance de ne pas dormir au bout de 10 minutes, mieux vaut connaître les règles du poker [...]*

8. Expressions can also be simple language collocations, i.e., multiple words that jointly have a different meaning than the ones of the individual components, and correspond to a conventional (and thus recurring) way of saying something. Often, the expression can be captured by using word n-grams as features (e.g., Dave et al. 2003). In the example below "at best" increases the negative feeling of the sentence (contrary to "best" in isolation).
   E.g., *Casino Royale finally hits full-throttle in its second hour but Bond fans will find the movie hit-and-miss **at best**.*

9. Single words can have different meanings depending on their use or the context in which they are used. In the first example, "cheap" is clearly negative, meaning "inferior"; in the second "cheap" means "not expensive".
   E.g., *Not a coincidence–GM used Mercedes' supplier for the new ... the interior plastics*

and wood trims is REALLY cheap. ... brown seats in a light colored car only make ...
E.g., *A Ferrari is not cheap to buy or run and residual values weaken if you use the car regularly.*

10. Other language specific problems regard the use of compounds in Dutch and the spelling in French. For instance, the Dutch "allersaaiste" translates to "uttermost boring", in which "boring" is not detected by our techniques. In some French examples, accents are omitted on "drôle" and "esthétique" in an inconsistent way.

Many errors relate to the lack of patterns in the training set and the incorrect handling of language phenomena. The former can be resolved by adding more training examples or using the machine learning techniques to select relevant examples to annotate, which can be successfully done as is shown in the next section. The latter can perhaps be improved by incorporating more language specific features or rules, with the restriction that the necessary resources might not be available for highly specific community blog languages. Some of the above types of errors demonstrate that an F-measure of 100% is out of reach only considering isolated sentences. We often lack additional context such as information found in previous sentences, knowledge of the subject domain or background knowledge of the readers of the text. When the sentiment classification is used in retrieval, the errors will hurt both its recall and precision.

Some of these problems have been noted by previous researchers. For example, the lack of world knowledge mentioned in 6. might be resolved by an approach like the one taken by Liu et al. (2003), who use the "Open Mind Common Sense" knowledge base to obtain large-scale real-world knowledge about people's common affective attitudes toward situations, things, people, and actions. For more references on some of these problems and their attempted solutions, see Section 2.

6.7 Active Learning

Recall from Section 3.5 that the goal of active learning is to reduce the number of examples that need to be labeled, without hurting performance. To this end, our tests aim at the detection of informative, similar and diverse examples.

6.7.1 Test Setup

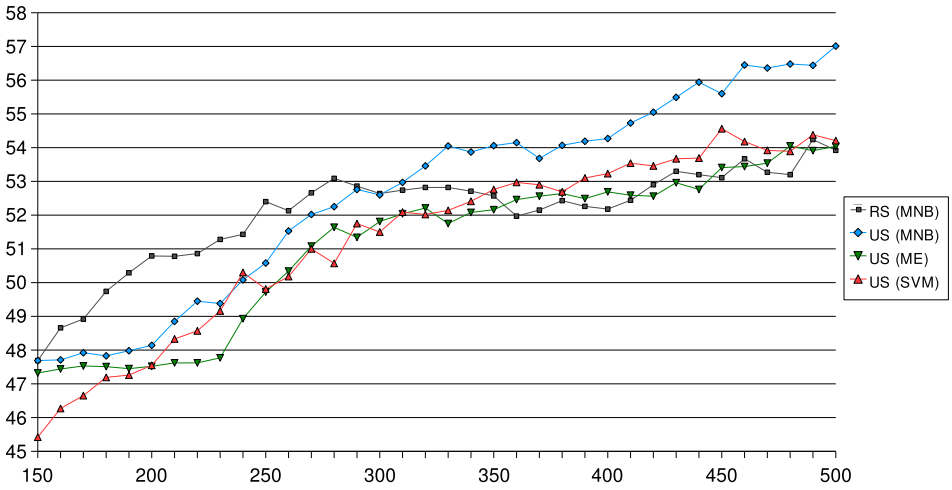
In all our approaches we rely on a good seed set that is already manually annotated in order to start from sufficient knowledge for the chosen active learning techniques to function properly. The size of this seed set is taken between 10 and 150 examples in our tests. We used a fixed corpus size of 500 examples as stopping criterion; the number of iterations depend on the batch size used. We performed our tests on the English corpus described above. The features used were unigrams augmented with negation tagging and discourse processing. The classifier used for evaluation was MNB, unless stated otherwise. Tests were performed on the corpus without removal of near duplicate examples (i.e., examples that are an exact subpart of another example), as this situation will also happen in reality, unless mentioned differently. To account for the effect of the seed set of examples, most tests were done five times using a different (semi-)randomly selected seed set, and the average was taken over these test runs. The same seeds (in practice, integers in the range [0, 4] that determine which examples are selected) for random selection will be used between different tests, making their results comparable. The standard active learning approach is always followed: in each sampling iteration, one example or a bucket of examples is selected. The labels will then be obtained automatically (as the examples are already labeled by an expert) and the examples are added to the training set. Until an example is selected, its label is not known to the active learning method.

We selected a corpus randomly from the sources cited above, of which one third was designated as our validation set (1703 examples: 1151 neutral, 274 positive, 72 negative), the remaining two thirds being the pool of unlabeled examples (2844 examples: 2296 neutral, 450 positive, 98 negative) from which also the seed sets are selected.

Table 10: Comparison of active learning results for the classifiers MNB, ME and SVM. #Ex stands for number of examples selected, US stands for uncertainty sampling and RS for random sampling, which is the baseline. Numbers for the active learners are the average F-measures over all classes. Numbers in **bold** indicate a significant ($\geq$ 97.5%) improvement over RS for the same number of examples selected – averaged over 5 runs.

	MNB		**ME**		**SVM**	
#Ex	**RS**	**US**	**RS**	**US**	**RS**	**US**
150	47.69	47.69	47.32	47.32	45.82	45.82
200	50.79	48.14	49.54	47.52	46.94	47.55
250	52.40	50.58	51.89	49.72	46.83	49.81
300	52.64	52.60	52.38	51.81	47.55	**51.50**
350	52.57	54.06	51.90	52.16	47.86	**52.76**
400	52.18	54.27	52.32	52.69	48.27	**53.23**
450	53.11	**55.60**	52.61	53.41	48.87	**54.56**
500	53.92	**57.01**	52.08	54.04	48.55	**54.21**

Figure 2: Graph representation of the average F-measure (Y-axis) in function of the number of documents selected (X-axis) by random sampling using an MNB classifier or uncertainty sampling using a specific classifier (cf. Table 10).



6.7.2 Informativeness

We perform the testing with a seed size of 150 examples and batch size of 10 examples, performing 35 iterations going up to 500 examples. For all classifiers (i.e., SVM (linear kernel), MNB and ME classifier) uncertainty sampling (US) results in a better performance in terms of F-measures averaged over the classes positive, negative and neutral as is seen in Table 10. The US results are plotted in Fig. 2. In US the number of near duplicate examples selected on average was 16.2/350, or 4.6%, compared to an estimated 12% in the entire corpus. In addition, the number of neutral examples selected on average was 234/350, or 66.9%, compared to 79.6% of neutral examples in the pool. The number of negative examples selected on average was 27.8/350, or 7.3%, compared to 3.7% of negative examples in the pool.

We see that the performance when using an SVM classifier seems to be already stagnating upon reaching 500 examples. An explanation might be that for problems which are not separable by the model being learned, US tends to select examples in the "mixture zones" between classes (Bondu et al. 2007). These examples hold little new information, but instead serve the

Table 11: Comparison of RS and US for the MNB uncertainty sampling method using seed size 150 and batch size 10. The number after ± is the standard deviation – averaged over 5 runs.

	Accuracy		**F-measure pos**		**F-measure neg**	
#Ex	**RS**	**US**	**RS**	**US**	**RS**	**US**
150	68.10±00.39	68.10±00.39	35.05±06.70	35.05±06.70	26.64±03.21	26.64±03.21
200	73.45±01.01	70.23±00.60	36.50±08.75	33.74±08.32	30.97±02.32	27.67±03.06
250	75.88±01.20	74.25±01.36	37.41±09.15	35.02±07.98	33.43±01.40	31.46±03.55
300	77.53±00.88	76.74±01.61	36.96±10.48	37.91±02.95	33.65±02.75	33.20±04.99
350	78.40±01.06	77.79±01.46	38.63±09.60	40.51±03.10	31.30±06.08	34.47±07.12
400	78.46±00.71	78.25±01.59	38.26±10.33	41.06±02.17	30.52±06.44	34.38±06.30
450	79.21±00.98	79.42±01.27	39.30±06.95	42.08±03.98	31.87±05.94	36.62±05.24
500	79.54±00.70	80.06±01.04	40.15±06.19	44.40±03.63	33.30±05.40	38.21±05.97

purpose of fine-tuning the current decision boundary of the classifier (Baram et al. 2003). For sentiment classification we expect the mixture zones to be well populated by examples that are either weakly opinionated or contain a mixture of positive and negative opinion words (accounted for as "ambiguous examples" in Section 6.6).

In Table 11, more detailed results for the MNB classifier in the above setup are shown including the burn-in period of the uncertainty sampling. Over the iterations precision steadily increases for both positive (up to 16%) and negative (up to 28%) classes, while precision of the neutral class only slightly decreases. Recall on the other hand improves for the positive (up to 4%) and neutral (up to 16%) classes, while it nearly halved for the negative (down 29%) class. These observations are easily explained as in each iteration relatively more positive and especially neutral examples are selected. Results for SVM and ME display the same trends, with the exception of recall of the positive class for ME, which decreases. The standard deviations give an indication of the difficulty of selecting "good" examples in each step of the iteration.

6.7.3 Similarity

Here we aim at obtaining negative examples to be annotated by a human as our sources contain on average less negative examples to train from. We took a realistic initial set of 80 examples: 40 neutral, 20 positive and 20 negative ones.

When running 50 iterations of relevance sampling with batch size 1, on average 31.8/50 examples selected by an MNB classifier were negative, being 63.6% compared to the 4.3% we would expect using random sampling. Using an SVM classifier this figure was 16.4/50 or 32.8%.

In a second experiment KCF (RBF kernel) was used to select examples closest to the initial 20 negative examples. When running again 50 iterations with batch size 1, on average 5.6/50 or 11.2% of the examples selected were negative, compared to 4.3% when we would select randomly. When looking only at the selected examples in the first 5 iterations, on average 4.2/5 were negative.

6.7.4 Diversity

When testing Kernel Farthest First and Kernel Average Farthest First (both RBF kernels) we start from 10 initial examples, as a good starting set should not be needed for this active learning method. The results can be seen in Table 12.

Both methods perform considerably worse than random sampling. KAFF performs better than KFF due to a higher F-measure for the positive class. Our assumption that KFF is a good means for bootstrapping a collection of examples does not hold on the corpus used. This is explained by the large quantity of diverse neutral examples, that may contain about any information and are found by this method. The percentage of neutral examples selected for KFF and KAFF is respectively 81.5% and 82.4%.

In the case of KFF, the number of redundant or semi-redundant sentences selected is an average

Table 12: Comparison of diversity results. KFF stands for kernel farthest first, KAFF for kernel average farthest first and RS for random sampling, which is the baseline. Evaluation was done using an MNB classifier. Numbers for the active learners are the average F-measures over all classes – averaged over 5 runs.

#Ex	RS	KFF	KAFF
10	32.53	32.53	32.53
20	39.38	36.68	37.00
50	39.90	38.34	38.34
100	39.05	39.54	41.00
150	42.30	38.20	40.42
250	44.29	39.85	40.98
500	49.28	42.83	44.27

of 3/490 or 0.6%, so indeed very low as we expected. However, for KAFF this number was 32/490 or 6.5%. This indicates a potential use for KFF in avoiding the selection of very similar examples.

An alternative approach for achieving diversity consists of training an SVM (linear kernel) for binary classification (neutral versus non-neutral) and selecting the 10 examples farthest from the hyperplane in each iteration; 5 on either side. The further settings used are identical to those described in 6.7.2. The number of positive and negative examples selected on average was 112.8/490, compared to 92.8/490 in the case of US (using an SVM). However, the results when training a three class classifier on the selected sentences actually become worse at each iteration, indicating that the found examples do not improve the decision boundaries of the SVM.

6.7.5 Combination

As an example experiment, we combine two active learning techniques, i.e., uncertainty and relevance sampling in a small algorithm (Algorithm 1).[20]

Algorithm 1 Active learning algorithm combining uncertainty sampling and relevance sampling, preceded by a fixed period of random sampling.

1: select examples randomly in buckets of 10 until they contain at least 4 neutral, 3 positive and 3 negative ones; label each bucket and retrain
2: select extra examples randomly up to 160 in buckets of ten (including the ones of the previous step); label each bucket and retrain
3: $newFmeasure \leftarrow validate()$
4: **repeat**
5: $\quad prevFmeasure \leftarrow newFmeasure$
6: $\quad$ select 160 examples in buckets of 10 using uncertainty sampling; label each bucket and retrain
7: $\quad$ select 10 examples using relevance sampling; label each bucket and retrain
8: $\quad newFmeasure \leftarrow validate()$
9: **until** $newFmeasure < prevFmeasure$

The results after the labeling of 500 examples are shown in Table 13. The combination of AL techniques is compared with a pure uncertainty sampling and pure random sampling (where selection is also done in buckets of ten, labeling of the buckets and retraining). From this experiment we learn that attempting to better balance the distribution of the classes in the training examples at regular intervals (here to obtain more negative examples by means of relevance sampling) has a positive effect. An initial random sampling has the extra advantage that we can make assumptions

[20] In the first selection step, taking into account the distribution of our dataset, the required examples are obtained (confidence level $\geq$ 99.9%) when we have seen 88 examples.

Table 13: Comparison of results when using Algorithm 1 with a pure uncertainty sampling (US) and pure random sampling (RS) approach. Numbers are the average F-measures over all classes – averaged over 5 runs.

#Ex	Algorithm 1	US	RS
150	39.91	41.88	42.30
200	39.84	43.98	43.05
250	44.25	44.53	44.29
300	46.05	45.04	44.89
350	48.06	47.85	46.20
400	48.94	48.53	47.45
450	50.91	50.81	48.85
500	53.36	51.38	49.28

about the distribution of the classes in our dataset, allowing to choose the appropriate relevance sampling approach.

7 Ideas for Future Improvements

Because the language in the corpora that we use for sentiment analysis is very varied, many of our errors relate to the lack of training examples. We explored techniques of active learning to optimize the annotation. We can, however, address this problem in several additional ways.

First the problem of *lack of training data* is especially apparent when dealing with "noisy" blog and forum texts, which diverge freely from formal language. At the level of preprocessing we could generalize the texts towards a kind of formal language (cf. De Smet and Moens 2007). Results of experiments using automatic spelling correction by Mullen and Malouf (2006) were inconclusive and further investigation might be useful. When dealing with formal language, collapsing some features may help to overcome data sparseness. We could consider a WordNet-based approach to collapsing features by mapping sets of near-synonyms to their common hypernym, but this leads to some practical problems posed by e.g., word sense ambiguity (words used in one sense may be mapped to the hypernym of another) and poor spelling in many blog texts. Fine tuning would be necessary so that the target words in the mapping are less specific but not too general, but we lack here a reliable automated approach. Alternatively, the application of rules that transform annotated examples in reliable syntactic variants seem useful.

Secondly, we might also make improvements or additions to the machine learning framework. We only tailored one example of a cascaded architecture in our experiments. Many other configurations of algorithms, features extractions and compositions of training sets can be integrated in the search for optimal classification models and thus optimal results. At each stage of the cascade we compute a probability or uncertainty by which each class is assigned, but when going deeper in the cascade, we never reuse these probabilities. A final classification can be computed conditioned on previous classifications of the example, eventually taking into account hidden variables that assess the importance of each layer in the cascade. Alternatively, we could look into semi-supervised techniques, which can further reduce the human effort required (e.g., along the lines of Zagibalov and Carroll 2008, who use a form of self-training for learning the polarity of lexical items).

We could think of integrating also a symbolic classifier into the cascade, for instance as a very first layer. However, a symbolic approach is generally deterministic, because it is based on string matching. The hand-crafted patterns do not catch all variations of natural language (e.g., with regard to negation), neither the meanings of words in different contexts, and neither the coupling of a sentiment to an entity or its attribute, so it is very likely that early in the cascade incorrect decisions are made. Alternatively a symbolic classification model could be used at the bottom of a cascade: difficult cases that cannot be classified with sufficient certainty are inspected by human annotators and the detected patterns might be added in a symbolic dictionary. Such an

approach would be in line with the technique of active learning where humans manually inspect and annotate sentences that are classified by the current classifier with low confidence.

A difficult problem is when *conflicting sentiments are expressed towards the entity of interest in one sentence*. Measuring the prevailing sentiment is a difficult task for a machine and is also for humans a subjective activity. It should be considered to assign a dual feeling to a sentence and its entity, or to assign a feeling to a specific attribute of the entity.

The expansion of our approach to new languages is substantial, but is a manageable task, due to the fact that our methods are largely language independent. Apart from the valuable rules for negation and discourse processing, parsing tools can optionally be used, when available.

Our results regard sentiment recognition in the car and movie domain. Our approaches should be tested and evaluated in other domains.

Finally, the extracted sentiments can be valuable when we aggregate them over time and across forums to yield a dynamic picture of what people think about certain products or persons (cf. Tong and Yager 2006). Such aggregated counts also require the consideration of all sentiment information that relates to the entity under consideration. This demands to accurately resolve all noun phrase coreferents (i.e., pronouns, synonyms) that refer to the entity that is monitored in the texts, which for noisy blogs poses additional difficulties.

8 Conclusions

We performed a large number of experiments to gain insights into sentiment classification of sentences or statements in blogs, consumer reviews and news forums, written in English, Dutch and French. The problem was to accurately classify the sentences according to the sentiment classes "positive", "negative" and "neutral" that is expressed with regard to an entity of interest in a setting where a rather limited number of annotated training examples are available and the texts are often not well-formed. The integrated approach combining methods from information retrieval, natural language processing and machine learning yielded good results given the difficulty of the task.

It was found that unigram features augmented with a limited number of language-specific features yield accuracy results of ca. 83%, 70% and 68% when classifying the English, Dutch and French Web data, respectively, according to sentiment, and slightly improve a baseline classification which only uses unigrams. A cascaded approach that reserves the computation of expensive features to a subset of the sentences further down in the cascade on its turn could only slightly positively influence accuracy and F-measures, but allowed to test a number of hypotheses. Among them, we found that the performance is increased by first filtering neutral sentences when sufficient training examples are available, before applying the sentiment classification algorithms. In the literature this first filtering was found useful when classifying complete review documents, and our tests confirm this finding for classifying individual sentences. Incorporating a layer in the cascade where expensive parse features are used, improved the performance for classifying sentences in which sentiments are expressed towards different entities.

Sparsity of training examples was an important cause of errors, which is especially severe in case the language of the sentences diverges largely from formal language (as is the case for French blogs). Labeling training examples is a tedious task, but techniques of active learning can certainly contribute here. Active learning, especially when combining several methods, provides a small, but noticeable improvement in average F-measures over randomly selecting examples for labeling. This makes it possible to arrive to similar results while annotating less examples, or to obtain better results when labeling the same amount. Another beneficial property of active learning is the reduction in the selection of redundant examples and the search for examples of a specific sentiment class. We observed that data that was crawled from the Web contains many duplicate or near-duplicate examples. As the sentiment analysis might be extended to include new domains and different languages, even a small benefit here is very valuable.

9 Acknowledgments

We are very grateful to the IWOIB (Institute for the Encouragement of Scientific Research and Innovation of Brussels) and the company Attentio, Belgium, who sponsored this research. We thank the Attentio team being Per Siljubergsåsen, Simon McDermott, Roxana Angheluta, Casper Davies and Frank Platteau for their valuable advise during multiple discussions and for the annotation of part of our data. We thank Gertjan van Noord (University of Groningen) who supplied the parser for Dutch. We are especially very grateful to Didier Bourigault (IRIT, France) for the parsing of the French sentences and to Tem Bertels who collected the French corpus and helped with the annotation of its sentences. Finally, we thank the reviewers for their very constructive comments.

References

Aman S and Szpakowicz S (2008). Using Roget's thesaurus for fine-grained emotion recognition. In: Proceedings of the International Joint Conference on NLP (IJCNLP). pp. 296–302.

Aue A and Gamon M (2005). Customizing sentiment classifiers to new domains: a case study. Technical report, Microsoft Research.

Bai X, Padman R and Airoldi E (2005). On learning parsimonious models for extracting consumer opinions. In: Proceedings of HICSS-05, 38th Annual Hawaii International Conference on System Sciences. IEEE Computer Society, Washington, DC, pp. 75–82.

Baram Y, El-Yaniv R and Luz K (2003). Online choice of active learning algorithms. In: Proceedings of ICML-03, 20th International Conference on Machine Learning. AAAI Press, Washington, DC, pp. 19–26.

Berger A L, Pietra S D and Pietra V J D (1996). A maximum entropy approach to natural language processing. Computational Linguistics, 22:39–71.

Bondu A, Lemaire V and Poulain B (2007). Active learning strategies: A case study for detection of emotions in speech. In: Industrial Conference on Data Mining, volume 4597 of Lecture Notes in Computer Science. Springer, pp. 228–241.

Brinker K (2003). Incorporating diversity in active learning with support vector machines. In: Proceedings of ICML-03, 20th International Conference on Machine Learning. AAAI Press, Washington, DC, pp. 59–66.

Budanitsky A and Hirst G (2004). Evaluating WordNet-based measures of lexical semantic relatedness. In: Computational Linguistics, volume 1. pp. 1–49.

Chambers N, Tetreault J, and Allen J (2006). Certainty identification in texts: Categorization model and manual tagging results. In: Shanahan J, Qu Y and Wiebe J, eds., Computing attitude and affect in text: Theory and applications. Springer, pp. 143–158.

Chesley P, Vincent B, Xu L and Srihari R (2006). Using verbs and adjectives to automatically classify blog sentiment. In: Proceedings of AAAI-CAAW-06, the Spring Symposia on Computational Approaches to Analyzing Weblogs. Stanford, CA.

Conrad J G and Schilder F (2007). Opinion mining in legal blogs. In: Proceedings of ICAIL '07, 11th International Conference on Artificial Intelligence and Law. ACM, New York, NY, USA, pp. 231–236.

Cristianini N and Shawe-Taylor J (2000). An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, Cambridge, UK.

Croft W and Lafferty B (2003). Language Modeling for Information Retrieval. Kluwer Academic Publishers, Boston, MA.

Dagan I and Engelson S P (1995). Committee-based sampling for training probabilistic classifiers. In: Proceedings of ICML-95, 12th International Conference on Machine Learning. pp. 150–157.

Dave K, Lawrence S and Pennock D M (2003). Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In: Proceedings of WWW-03, 12th International Conference on the World Wide Web. ACM Press, New York, pp. 519–528.

De Smet W and Moens M F (2007). Generating a topic hierarchy from dialect texts. In: DEXA Workshops. IEEE Computer Society, pp. 249–253.

Finn A and Kushmerick N (2003). Learning to classify documents according to genre. Journal of the American Society for Information Science, 57:1506–1518. Special issue on Computational Analysis of Style.

Freund Y, Seung H S, Shamir E and Tishby N (1997). Selective sampling using the query by committee algorithm. Machine Learning, 28:133–168.

Galley M and McKeown K (2007). Lexicalized Markov grammars for sentence compression. In: Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference. Association for Computational Linguistics, Rochester, New York, pp. 180–187.

Gamon M (2004). Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In: Proceedings of COLING-04, the 20th International Conference on Computational Linguistics. Geneva, CH, pp. 841–847.

Hatzivassiloglou V and McKeown K R (1997). Predicting the semantic orientation of adjectives. In: Proceedings of ACL-97, 35th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Madrid, Spain, pp. 174–181.

Hatzivassiloglou V and Wiebe J M (2000). Effects of adjective orientation and gradability on sentence subjectivity. In: Proceedings of COLING-00, 18th International Conference on Computational Linguistics. Morgan Kaufmann, San Francisco, CA, pp. 299–305.

Hearst M A (1992). Direction-based text interpretation as an information access refinement. In: Jacobs P, ed., Text-based intelligent systems: current research and practice in information extraction and retrieval. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, pp. 257–274.

Hu M and Liu B (2004). Mining opinion features in customer reviews. In: Proceedings of AAAI-04, 19th National Conference on Artificial Intellgience. San Jose, US, pp. 755–760.

Huang T, Dagli C, Rajaram S, Chang E, Mandel M, Poliner G and Ellis D (2008). Active learning for interactive multimedia retrieval. Proceedings of the IEEE, 96:648–667.

Huber R, Batliner A, Buckow J, Nöth E, Warnke V and Niemann H (2000). Recognition of emotion in a realistic dialogue scenario. In: Proceedings of the International Conference on Spoken Language Processing, volume 1. Beijing, China, pp. 665–668.

Iyengar V S, Apte C and Zhang T (2000). Active learning using adaptive resampling. In: Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, pp. 92–98.

Kamps J and Marx M (2002). Words with attitude. In: Proceedings of the 1st International Conference on Global WordNet. Mysore, India, pp. 332–341.

Kessler B, Nunberg G and Schütze H (1997). Automatic detection of text genre. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, Somerset, New Jersey, pp. 32–38.

Knight K and Marcu D (2000). Statistics-based summarization - step one: Sentence compression. In: Proceedings of AAAI/IAAI-00, 12th Conference on Innovative Applications of AI. AAAI Press, San Francisco, CA, pp. 703–710.

Kobayashi N, Inui K and Matsumoto Y (2007). Extracting aspect-evaluation and aspect-of relations in opinion mining. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). pp. 1065–1074.

Leshed G and Kaye J (2006). Understanding how bloggers feel: recognizing affect in blog posts. In: CHI '06: Extended Abstracts on Human Factors in Computing Systems. ACM, New York, pp. 1019–1024.

Lewis D D and Catlett J (1994). Heterogeneous uncertainty sampling for supervised learning. In: Proceedings of ICML-94, 11th International Conference on Machine Learning. Morgan Kaufmann Publishers, San Francisco, CA, New Brunswick, US, pp. 148–156.

Lewis D D and Gale W A (1994). A sequential algorithm for training text classifiers. In: Proceedings of SIGIR '94, 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Springer-Verlag, New York, pp. 3–12.

Liere R and Tadepalli P (1997). Active learning with committees for text categorization. In: Proceedings of AAAI-97, 14th Conference of the American Association for Artificial Intelligence. AAAI Press, Menlo Park, CA, pp. 591–596.

Liu H, Lieberman H and Selker T (2003). A model of textual affect sensing using real-world knowledge. In: Proceedings of IUI-03, 8th International Conference on Intelligent User Interfaces. ACM, New York, pp. 125–132.

Manning C D, Raghavan P and Schütze H (2008). Introduction to Information Retrieval, chapter 13. Cambridge University Press.

McCallum A K and Nigam K (1998). Employing EM in pool-based active learning for text classification. In: Proceedings of ICML-98, 15th International Conference on Machine Learning. Morgan Kaufmann Publishers, San Francisco, CA, pp. 350–358.

Mishne G (2005). Experiments with mood classification in blog posts. In: Style2005, 1st Workshop on Stylistic Analysis of Text for Information Access at SIGIR 2005.

Mishne G and de Rijke M (2006). A study of blog search. In: European Conference on Information Retrieval. Springer, Berlin, Germany, pp. 289–301.

Mulder M, Nijholt A, den Uyl M and Terpstra P (2004). A lexical grammatical implementation of affect. In: Proceedings of TSD-04, 7th International Conference on Text, Speech and Dialogue, volume 3206 of Lecture Notes in Computer Science. Springer, Berlin, Germany, pp. 171–178.

Mullen T and Collier N (2004). Sentiment analysis using support vector machines with diverse information sources. In: Proceedings of EMNLP-04, 9th Conference on Empirical Methods in Natural Language Processing. Barcelona, Spain, pp. 412–418.

Mullen T and Malouf R (2006). A preliminary investigation into sentiment analysis of informal political discourse. In: AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW 2006). pp. 125–126.

Nguyen H T and Smeulders A (2004). Active learning using pre-clustering. In: Proceedings of ICML-04, 21st International Conference on Machine learning. ACM Press, New York, p. 79.

Nijholt A (2003). Humor and embodied conversational agents. CTIT Technical Report series No. 03-03, University of Twente.

Osugi T (2005). Exploration-Based Active Machine Learning. Master's thesis, University of Nebraska.

Pang B and Lee L (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of ACL-04, 42nd Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, East Stroudsburg, PA, pp. 271–278.

Pang B, Lee L and Vaithyanathan S (2002). Thumbs up? Sentiment classification using machine learning techniques. In: Proceedings of EMNLP-02, the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Philadelphia, PA, pp. 79–86.

Pedersen T (2001). A decision tree of bigrams is an accurate predictor of word sense. In: Proceedings of the Second Annual Meeting of the North American Chapter of the Association for Computational Linguistics. pp. 79–86.

Polanyi L and Zaenen A (2006). Contextual valence shifters. In: Shanahan J, Qu Y and Wiebe J, eds., Computing attitude and affect in text: Theory and applications. Springer, pp. 1–10.

Porter M F (1980). An algorithm for suffix stripping. Program, 14:130–137.

Raina R, Battle A, Lee H, Packer B and Ng A Y (2007). Self-taught learning: transfer learning from unlabeled data. In: Proceedings of ICML-07, 24th International Conference on Machine Learning. ACM, New York, pp. 759–766.

Riloff E, Wiebe J and Wilson T (2003). Learning subjective nouns using extraction pattern bootstrapping. In: Proceedings of CoNLL-03, 7th Conference on Natural Language Learning. Edmonton, CA, pp. 25–32.

Roy N and McCallum A (2001). Toward optimal active learning through sampling estimation of error reduction. In: Proceedings of ICML-01, 18th International Conference on Machine Learning. Morgan Kaufmann, San Francisco, CA, pp. 441–448.

Rubin V L, Liddy E D and Kando N (2006). Certainty identification in texts: Categorization model and manual tagging results. In: Shanahan J, Qu Y and Wiebe J, eds., Computing attitude and affect in text: Theory and applications. Springer, Berlin, Germany, pp. 61–76.

Saar-Tsechansky M and Provost F (2004). Active sampling for class probability estimation and ranking. Machine Learning, 54:153–178.

Salvetti F, Lewis S and Reichenbach C (2004). Impact of lexical filtering on overall opinion polarity identification. In: Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications. Stanford, CA.

Seung H S, Opper M and Sompolinsky H (1992). Query by committee. In: Computational Learning Theory. pp. 287–294.

Tong R and Yager R (2006). Characterizing buzz and sentiment in internet sources: Linguistic summaries and predictive behaviors. In: Shanahan J, Qu Y and Wiebe J, eds., Computing attitude and affect in text: Theory and applications. Springer, pp. 281–296.

Tong S and Koller D (2002). Support vector machine active learning with applications to text classification. Journal of Machine Learning Research, 2:45–66.

Turney P (2002). Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: Proceedings of ACL-02, 40th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Philadelphia, PA, pp. 417–424.

Viola P and Jones M (2001). Robust real-time object detection. Technical report, Cambridge Research Lab, Compaq.

Wang B and Wang H (2007). Bootstrapping both product properties and opinion words from Chinese reviews with cross-training. In: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence. IEEE Computer Society, Washington, DC, USA, pp. 259–262.

Whitelaw C, Garg N and Argamon S (2005). Using appraisal taxonomies for sentiment analysis. In: Proceedings of MCLC-05, 2nd Midwest Computational Linguistic Colloquium. Columbus, OH.

Wiebe J (2000). Learning subjective adjectives from corpora. In: Proceedings of AAAI-00, 17th Conference of the American Association for Artificial Intelligence. AAAI Press / The MIT Press, Austin, TX, pp. 735–740.

Xu Z, Yu K, Tresp V, Xu X and Wang J (2003). Representative sampling for text classification using support vector machines. In: European Conference on Information Retrieval. Springer, Berlin, Germany, pp. 393–407.

Zagibalov T and Carroll J (2008). Unsupervised classification of sentiment and objectivity in Chinese text. In: Proceedings of the International Joint Conference on NLP (IJCNLP).

Zhang E and Zhang Y (2006). UCSC on REC 2006 blog opinion mining. Technical report, University of California Santa Cruz, CA.

Zhu J, Wang H and Hovy E H (2008). Learning a stopping criterion for active learning for word sense disambiguation and text classification. In: Proceedings of the International Joint Conference on NLP (IJCNLP).