Enhancing Sentiment Extraction from Text by Means of Arguments *

Lucas Carstens
Imperial College London
South Kensington Campus
Huxley Building, Room 429
London SW7 2AZ, UK
lucas.carstens10@imperial.ac.uk

Francesca Toni
Imperial College London
South Kensington Campus
Huxley Building, Room 430
London SW7 2AZ, UK
ft@imperial.ac.uk

ABSTRACT

Sentiment Analysis is concerned with (1) differentiating opinionated text from factual text and, in the case of opinionated text, (2) determine its polarity. With this paper, we address problem (1) and present A-SVM (Argument enhanced Support Vector Machines), a multimodal system that focuses on the discrimination of opinionated text from non-opinionated text with the help of (i) Support Vector Machines (SVM) and (ii) arguments, acquired by means of a user feedback mechanism, and used to improve the SVM classifications. We have used a prototype to investigate the validity of approaching Sentiment Analysis in this multi faceted manner by comparing straightforward Machine Learning techniques with our multimodal system architecture. All evaluations were executed using a purpose-built corpus of annotated text and A-SVM's classification performance was compared to that of SVM. The classification of a test set of approximately 4,500 n-grams yielded an increase in classification precision of 5.6%.

Keywords

Sentiment Analysis, Support Vector Machines, Argumentation, User feedback, A-SVM

1. INTRODUCTION

Today, more than ever, the World Wide Web offers unprecedented opportunities to equally produce and consume data and information. At the same time, the immense pool of content emerging from a collaborative environment such as the Web carries significant implications when it comes to putting this content to use. Sentiment Analysis, or Opinion Mining, attempts to (1) differentiate opinionated text from factual text and, once text is deemed to be opinionated, (2) classify it as expressing a negative, neutral or positive opinion (see [18, 29] for an overview). The approach to Sentiment Analysis presented in this paper adds arguments, as understood in [23] and explained in section 2, to the probabilistic method of Support Vector Machines (SVM) [5] in order to distinguish opinionated from non-opinionated text and thus address problem (1) above. We have developed A-SVM, a *multimodal system* that classifies text according to its opinionatedness by means of combining SVM and arguments. We refer to A-SVM as multimodal because the system architecture incorporates concepts from both Machine Learning and Argumentation Theory. We have also developed a text corpus, consisting of approximately 13,000 annotated n-grams (each n-gram is part of a larger sentence) that are represented as feature vectors. In a preliminary evaluation, comparing A-SVM with a straightforward SVM classifier on our corpus, we achieved an increase in classification precision of 5.6%, from 78.1% to 83.7%, a 2.41% increase in recall and 4.05 points for the F1 measure.

Our approach can be seen as a novel way to integrate reasoning with rules (in the form of simple arguments) within probabilistic methods. Thus, we do not solely present a novel approach to Sentiment Analysis, but also investigate the merits of interweaving fields that have previously exhibited limited common ground. The main hypothesis supporting our approach is that combining substantially different methods of of dealing with written language computationally should allow an increase of general performance compared to applying each method in isolation. Our encouraging experimental results corroborate this hypothesis.

The paper is organised as follows: In section 2, we give an overview of the techniques used for the development and execution of A-SVM. In section 3 we introduce the corpus that forms the basis for text classification in our system, followed by A-SVM, the system itself, in section 4. In section 5 we present a twofold evaluation of the system, one quantitive, the other qualitative. The evaluation is followed by related work in section 6 and conclusions in section 7.

2. BACKGROUND

Our system makes use of SVM and arguments. Moreover, it is trained on a purpose built text corpus. In this section we first present the sources utilised in developing the corpus (section 2.1). We then go on to review SVM (section 2.2) and Argument Based Machine Learning (ABML) (section 2.3) from which we draw inspiration on how one may integrate arguments in Machine Learning.

*An extended version of this paper is available at `www.doc.ic.ac.uk/~lc1310`

WISDOM '13, August 11 2013, Chicago, USA.

2.1 Resources

The corpus we have developed during our research, described in detail in section 3, is built from a number of sources. We describe each of them below:

- The *MPQA corpus* [32] is a collection of 378 news articles, comprising around 10,000 sentences, each of which has been manually annotated with tags describing a number of subjectivity measures, as well as sources and objects of opinions within the text. The annotation scheme, proposed by Wiebe and colleagues, used to develop the MPQA corpus annotates the texts at word and phrase level. It thus applies relatively fine grained information about the articles annotated. The corpus annotations include "various properties involving intensity, significance, and type of attitude" [32], as well as records of the source of a statement and who a statement is directed towards. The complexity of the annotations is exacerbated by the way the annotations were contrived. All annotations were done manually and the annotators were given annotation guidelines that were rather loose and left the annotators with large room of choice in their annotation. The result is a very intricately annotated corpus. Since, for our purposes, the level of detail provided by the MPQA corpus exceeds our needs, we only use those annotations that define pieces of text as either opinionated or not opinionated.

- *SentiWordNet* [14] is a lexical resource that has been specifically designed for the purpose of aiding Sentiment Analysis, based on WordNet [20], but extended with lexical information about the sentiment of each synset contained in WordNet. A synset is a collection of words comprising all synonyms listed in WordNet for any particular word. The additional information provided by SentiWordNet, but not present in WordNet, comes in the form of three different values (positivity, objectivity and negativity) which sum to one and describe the orientation of sentiment.

- *TreeTagger* [31] is a publicly available Part-Of-Speech (POS) tagging system that uses decision trees for probabilistically selecting POS annotations. The aim of using decision trees in tagging POS is to allow taking into account the context in which the word or phrase that is tagged appears. The leaf nodes of the decision tree mark the actual decision on how to tag the word or phrase while the higher nodes give information about the surrounding words.

2.2 Support Vector Machines

Supervised Learning techniques have been the most prominently used techniques in Sentiment Analysis (see, for example, [16, 30]), with SVM yielding some of the most promising results (e.g. [24]). Here we briefly review SVM (see [5] for a more detailed introduction). Consider a linear model describing a binary classification problem such as the one of determining opinionatedness of a phrase. We can describe such a problem with a linear model of the form

$$y(x) = \mathbf{w}^T \phi(x) + b$$

Name	PaysReg.	Rich	HairColour	CreditAppr.
Mrs Brown	No	Yes	Blond	Yes
Mr Grey	No	No	Grey	No
Miss White	Yes	No	Blond	Yes

Table 1: Training Examples for ABCN2 learning algorithm

where $\phi(x)$ is a feature-space transformation of the data x (in our case a piece of text), $\mathbf{w}^T$ is a weight vector and b is a bias. Transforming data into feature space can yield linear separability of data that is not linearly separable in the original data space. A training data set consists of N input vectors $x_1, ..., x_N$, all of which have a class label $C \in \{1, -1\}$, and new data x is classified according to the sign of $y(x)$.

2.3 Argument-based Machine Learning

Argumentation (see [4] for an overview) can support the (dialectical) justification of conclusions. ABML [23] combines user feedback in the form of justified arguments and classical supervised machine learning (in the form of the CN2 algorithm [12]) to enhance performance. Usually supervised machine learning techniques take a preferably large number of training examples such as the ones used by Mozina and colleagues in [23], shown in table 1, and try to find a hypothesis that adequately explains the training examples and then correctly classifies new cases. In the example we have a number of parameters, e.g. *HairColour*, and a class label for each example, i.e. *CreditApproved*. Within the framework of ABML, some of these training examples have an associated argument explaining the reasoning behind why an example is classified the way it is. Consider again the example shown in table 1. The CN2 algorithm takes as input examples in the form of pairs (A, C), where A is an attribute-value vector, e.g. $(Name = MrsBrown, PaysRegularly = No, Rich = Yes, HairColour = Blond)$ and C is the class the example belongs to, e.g. $(CreditApproved = Yes)$. ABML accepts such examples, as well, but in addition is able to process examples of the form $(A, C, Arguments)$, where $Arguments$ is a set of arguments of one of the forms:

$$C \text{ because } Reasons \quad \text{or} \quad C \text{ despite } Reasons$$

where *Reasons* is a conjunction of attribute-value pairs such as

$$Arguments = \{C \text{ because } Rich = Yes,\\ C \text{ despite } PaysRegularly = No\}.$$

We will use arguments of analogous format but with n-grams as *Reasons* and classifications of (non-)opinionatedness as conclusions C.

3. THE TEXT CORPUS

We have developed a text corpus (available at `www.doc.ic.ac.uk/~lc1310`) of roughly 13,000 semi-automatically annotated n-grams, which was then used to train the SVM that classifies new text within A-SVM. The corpus was constructed using version 2.0 of the MPQA corpus, SentiWordNet and TreeTagger (see section 2.1), each of which contributed text, features annotating this text or both. Around

60% of the extracted n-grams were classified as opinionated and 40% as non-opinionated. Each n-gram in our corpus is associated with a vector of features describing certain characteristics of the n-gram. The maximum length of n-grams in the corpus is five words and all n-grams are extracted from the MPQA corpus. This value n$\leq$ 5 was chosen as a compromise between running into potential computational problems and hampering the ability to grasp the role that the context in which a word appears plays. The feature vector associated with an n-gram is comprised of up to nine features:

- one feature is the size of the n-gram;
- three features represent scores of positivity, neutrality and negativity extracted from the SentiWordNet lexicon;
- between one and five (given by n) additional features represent, for the words that appear in the n-gram, the words' lexical types and their basic form, which we obtain by applying TreeTagger to the n-gram.

In addition to the features, each annotated n-gram in our corpus contains a class label, $C \in \{+1, -1\}$, for the particular n-gram, identifying it as either *opinionated* ($C = +1$) or *non-opinionated* ($C = -1$). This class label is automatically extracted from the MPQA corpus, since the n-grams of the MPQA corpus are fully annotated with respect to opinionatedness. For example, our corpus contains the 2-gram *"refused to"*, annotated by

2 0.0 0.8125 0.1875
refused <VHZ >(refuse) to <TO >(to) +1

where n=2, positivity=0, neutrality=0.8125, negativity = 0.1875, followed by the POS tags and their basic forms and the classification +1, i.e. opinionated.

4. THE A-SVM SYSTEM

We present A-SVM, a multimodal system that tackles the discrimination of opinionated text from non-opinionated text with the help of SVM and a simple form of argumentation. A-SVM is, from a high-level perspective, comprised of a succession of input gathering, input conversion and input classification tasks. This succession is presented schematically in figure 1 and explained in detail throughout the remainder of this section. We refer to the *Activity* and *Data* nodes shown in figure 1 whenever the according part of the system is described below.

In a two-step classification process, SVM classifications (*"Preliminary classification"*) are improved via arguments, acquired by means of a user feedback mechanism, to obtain *"Final classification"*. The system has been trained on the corpus described above, but has been designed to analyse any textual input supplied by a user via a Graphical User interface (GUI) (*"GUI1: User input"*). We focus on the input conversion and input classification tasks. To describe the vital aspects of A-SVM we shall consider a single system execution from start to finish. Once initialised, the system prompts a GUI (*"GUI1: User input"*) to the user through which he or she provides and submits a piece of text, which we shall denote *TXT* subsequently. Let us assume the user has typed *TXT*: "*Despite the mounting pressure he has refused to bow*", taken from the MPQA corpus. *TXT* contains the 2-gram we have used as an example before, i.e. "*refused to*", and to describe the vital aspects of A-SVM we shall consider the n-gram (for n=4) "*has refused to bow*". This is an interesting n-gram because the word "*refused*" can be thought to be opinionated while we nevertheless argue that a clear opinionatedness of the 4-gram may be disputed.

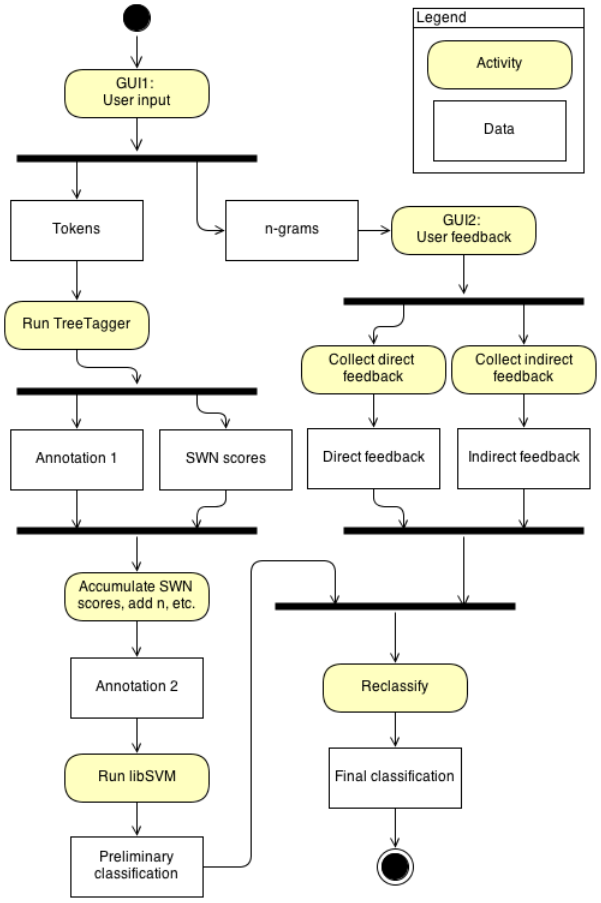


Figure 1: UML diagram showing a schematic view of A-SVM

Input conversion. Once the user has submitted the input (*TXT*) the first step in processing it is to break it up into single words (*"Tokens"*), each of which is annotated individually (*"Annotation 1"*). The entire annotation process takes place hidden from the user. The individual word annotations are then merged to form annotated n-grams (for all possible n-grams of up to five words in the given input). Each word is annotated with TreeTagger and with the scores extracted from the SentiWordNet lexicon, if the word occurs in the SentiWordNet lexicon, or with scores of zero, otherwise.

Resulting from this procedure are annotated single words

bearing five features each: scores for positivity, neutrality and negativity, the word's POS tag and its basic form. For example, the word "*has*" in our illustrative n-gram will be annotated as

0.0 1.0 0.0 has <VHZ >(have)

The next step is to construct annotated n-grams from the given input text, constructing them from the annotated single words. This step yields every possible annotated n-gram contained in the user input up to a length of five words. For any input text larger than three words, we obtain m=(n-2)*5 n-grams up to a size of five words. Each n-gram that is constructed is given one score for positivity, one for neutrality and one for negativity, obtained from the individual scores of the words in the n-gram, summed up and normalised by dividing by n. Lastly, each obtained n-gram is annotated with the size of the n-gram. For our example, the result of this procedure includes:

1 0.0 1.0 0.0 has <VHZ >(have)

2 0.0 0.8125 0.1875
has<VHZ >(have) refused<VVN>(refuse)

3 0.0 0.875 0.125
has<VHZ >(have) refused<VVN>(refuse) to<TO>(to)

4 0.0 0.78125 0.21875
has<VHZ >(have) refused<VVN>(refuse) to<TO>(to) bow<VV>(bow)

...

1 0.0 0.625 0.375 refused<VHZ >(refuse)

2 0.0 0.8125 0.1875
refused<VHZ >(refuse) to<TO>(to)

3 0.0 0.7083 0.2917
refused<VHZ >(refuse) to<TO>(to) bow<VV>(bow)

...

The final step before classifying *TXT* is a conversion of any non-numerical values in the annotations to numerical values, since SVM require strictly numerical inputs. This is done by simply assigning each possible POS tag a unique number between zero and one. This procedure produces what we refer to as *"Annotation 2"* in figure 1.

Preliminary classification. The SVM algorithm is applied to all annotated n-grams resulting from processing the input *TXT* as described above. This results in one class label, $C \in \{+1, -1\}$, per annotated n-gram, thus classifying each n-gram as being either opinionated (+1) or non-opinionated (-1) (*"Preliminary classification"*). The user is then presented with a second GUI (*"GUI2: User feedback"*) through which he or she is asked to provide the system with some (in the current version five) of their own classifications of n-grams, which are randomly selected from the original user input *TXT*. In addition to judging these n-grams as being opinionated or non-opinionated, the user gives a justification for this classification. This justification is given in the form of one argument per n-gram, in a syntax as described below.

Generating Arguments from user feedback. In giving arguments the user specifies the following:

1. The class label, $C \in \{+1, -1\}$, indicating whether the argument is in favour of the n-gram being *opinionated* or *non-opinionated*;
2. The direction of reasoning, i.e. *because* or *despite*;
3. The part of the n-gram that is *most* responsible for the user's judgement.

For example, a possible argument concerning the n-gram we have used before, "*has refused to bow*", may be

+1 (opinionated) because "refused to bow"

Here, *"refused to bow"* is the *Reason* for the conclusion $C = +1$ (see section 2.3). Thus, in this hypothetical scenario the user has judged the n-gram to be opinionated and identifies words two to four in the input as the reason for his or her judgement. In the same manner, the user provides four additional arguments by passing judgement upon four more randomly selected n-grams and providing reasons to support the decision. Below we refer to the five arguments given by the user as *ARGS*. The feedback is considered in two ways, as illustrated below.

Direct and indirect user feedback for reclassification. *ARGS* consists of classifications for five n-grams in *TXT* and reasons for those classifications. Disregarding *Reasons* leaves a classification of the five n-grams. This is the direct feedback, used to overrule the classification of the SVM for those particular n-grams. We base this overruling on the choice to trust the manual classification (and thus the user) over the SVM classification. In order to gain information from the user's arguments that goes beyond simply reclassifying five n-grams, as done with the direct feedback, we construct what we call *indirect feedback*. This results in the re-classification of a subset of all n-grams that can be obtained from the original user input *TXT*. This subset consists of all n-grams that have the same structure, in terms of POS tags, as the n-grams the user classified through *ARGS*. Though this beckons further investigation, our preliminary evaluations hint that certain POS tag combinations may hold information about (non-)opinionatedness of text. Assuming this holds true, having the tags as indicators for a class means that it suffices that an n-gram in the text contains the same combination of POS tags as the n-gram that was used to construct the argument, while the n-gram does not have to be exactly the same.

Algorithmically, the indirect feedback works as follows: Each n-gram obtained from the original user input (TXT) is assigned a score, initially set to zero. Then, for each such n-gram, we check whether it has the same POS tag combination as one of the *Reasons* the user has provided in $ARGS$. If one such *Reason* exists in an argument with conclusion C this n-gram's score is either increased by one (if $C = +1$) or decreased by one (if $C = -1$).

At the end of this process, the original classification, i.e. *"Preliminary classification"*, is overwritten according to the sign of the score (to opinionated if the score is > 1 and to non-opinionated if it is < -1). If the value we obtain for an n-gram is between -1 and 1 we simply retain the original classification provided by the SVM. From this reclassification procedure we obtain our *"Final classification"*.

Algorithm 1 Pseudocode describing the final classification procedure

```
1: let N = {n_0, ..., n_4} be the n-grams for user feedback
2: let U = {u_0, ..., u_4} be the user feedback class labels
3: let A = {a_0, ..., a_4} be the feedback reasons
4: let m be the total number of n-grams constructed from
   the user input
5: let F = {f_0, ..., f_m} be feature vectors representing the
   original user input's POS tags
6: let L = {l_0, ..., l_m} be the class labels determined for
   F = {f_0, ..., f_m} by SVM
7: let V = {v_0 = 0, ...v_m = 0} be the indirect feedback
   values for F = {f_0, ..., f_m}
8: counter ← 0
9: while counter < m do
10:   for i = 0 to 4 do
11:     if f_counter == n_i then
12:       l_counter ← u_i
13:     else if a_i ∈ f_counter then
14:       if l_counter == +1 then
15:         v_counter + +
16:       else
17:         v_counter − −
18:       end if
19:     end if
20:   end for
21:   counter + +
22: end while
23: for j = 0 to m do
24:   if v_j > 1 then
25:     l_j ← +1
26:   else if v_j < −1 then
27:     l_j ← −1
28:   end if
29: end for
```

Note that some preprocessing of $ARGS$ is needed before executing the scoring algorithm. An argument of the form

- *opinionated because Reasons* is converted to the pair $< +1, POS >$,
- *opinionated despite Reasons* is converted to the pair $< -1, POS >$,
- *non-opinionated because Reasons* is converted to the pair $< -1, POS >$,
- *non-opinionated despite Reasons* is converted to the pair $< +1, POS >$,

where POS denotes the POS tag combination that is found in *Reasons* and POS is associated with a class label, $C \in \{+1, -1\}$, that is derived from the combination of opinionated (non-opinionated) and because (despite) as shown. For our example n-gram *"refused to bow"* we would attain the POS tag combination *VVN-TO-VV* and thus (1) is mapped to $< +1, VVN - TO - VV >$.

Final classification. This is the result of combining evidence from SVM and arguments provided by the user where the classification results of the SVM form the basis, with the arguments overruling the SVM classification whenever the evidence supplied by them is deemed strong enough. The calculation of the final classification is summarised in algorithm 1.

Line 11 checks whether the current feature vector has the equivalent POS feature values as one of the n-grams classified by the user feedback. If this is the case the class label l_m is overwritten with the user's classification (line 12). If this is not the case, we check whether the combination the user chose as being responsible for his or her choice of classification is part of the current n-gram's features (line 13). If this is the case, we either increase or decrease the $v_{counter}$, depending on the class label of the sub n-gram (lines 14 to 17). Depending on the value of $v_{counter}$, we ultimately determine our confidence in the part of user input being either opinionated or non-opinionated. After all $F = \{f_0, ..., f_m\}$ have been processed, all the n-grams' class labels whose indirect feedback value v_j surpass the threshold are changed accordingly (lines 23 to 27).

Classification output. Once we have obtained the final classification all that is left to do is presenting A-SVM's classifications to the user. An example output is shown in figure 2. Each word from the user input TXT, i.e. *"Although I like sunny weather,..."*, is assigned a score between *-1* and *+1*. This score is captured as follows. Each word of TXT will appear in more than one n-gram and will thus be classified more than once. Starting from zero we sum all classification results for each word; whenever a word appears in an n-gram that is classified as opinionated we increase the score by one; when the n-gram is classified as non-opinionated, we decrease it by one. The final sum is normalised, giving a value between -1 and 1. A graph (as in figure 2) is shown to the user to represent this *"vote of confidence"* of the system as to how sure it is that a word, in the context of the input sentence, is either opinionated or non-opinionated.

The closer a value is to $+1$, the more confident the system is that this word is opinionated, the closer we get to -1, the higher the system's confidence in this word's being non-opinionated. When the value tends to 0, the system has received conflicting evidence about this particular word's

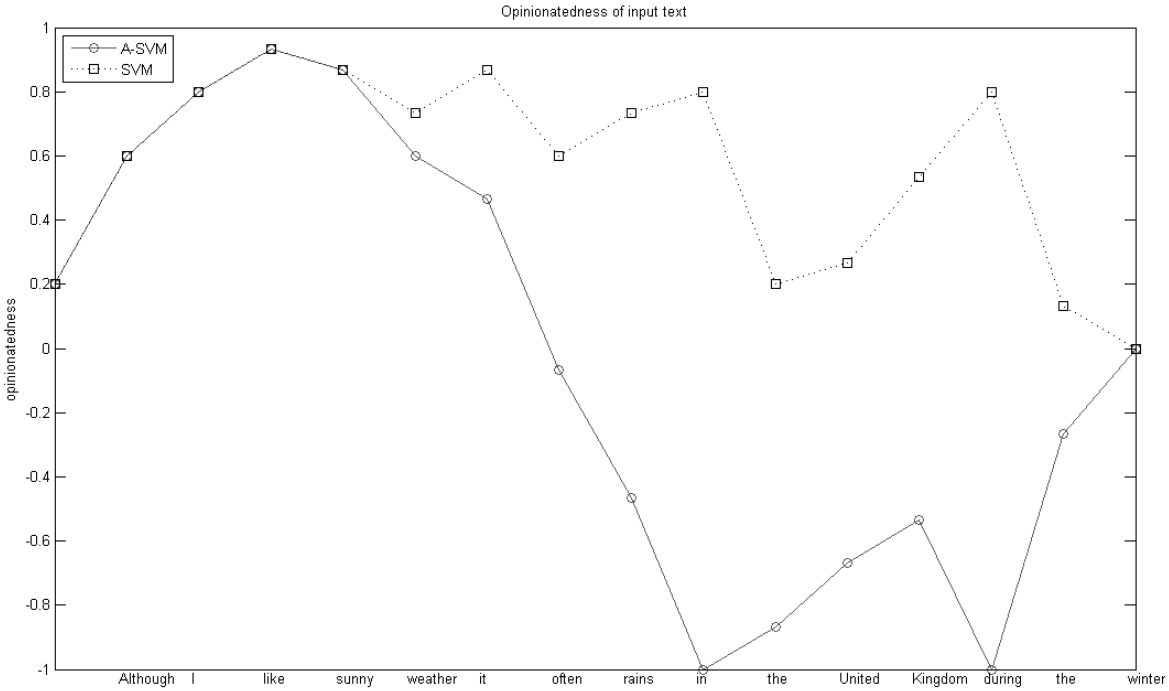


Figure 2: Classification results of an example user input for SVM and A-SVM

opinionatedness and is thus unable to identify a clear leaning. Figure 2 also shows the classification result (dotted line with boxes) using solely SVM, for the sake of comparison. As we can see, SVM arguably misclassifies most of the main clause ("it often rains...") whereas A-SVM seems to rectify this error.

5. SYSTEM EVALUATION

A number of supervised learning algorithms have been proposed to tackle different issues of Sentiment Analysis, some of which are compared in [30]. Pang and colleagues use three different classifiers for the task of classifying movie reviews at document level and compare the performance of Naive Bayes classifier, Maximum Entropy classifier and SVM. The results presented by Pang and colleagues show that, for the particular task of classifying movie reviews at document level, the best performance is achieved using SVM while the worst performing method is the Naive Bayes classifier. In line with these results we have focused on evaluating our system's performance using SVM.

The main question we are trying to answer is whether adapting a multimodal approach to Sentiment Analysis may prove valuable to the cause of furthering the development of systems. To do so, we put the classification performance of the developed system up against the classification performance of the SVM incorporated in the system, stripped bare of everything but the SVM itself. In addition to this comparison of classification performance, a user centred evaluation was conducted. This aimed at complementing the results of the quantitive evaluation with qualitative insights into the subjective impression users get from the system's performance.

Kernel	K. degree	gamma	K. coeff.	Cost	bias
linear	1	0.001	0	1	1
polynomial	to	to	to	to	to
RBF	10	100	1	60	10
sigmoid					

Table 2: parameter values tested for SVM training

5.1 SVM vs A-SVM

In order to evaluate the system's classification performance, the corpus of n-grams described in section 3 was split into a training set and a test set: 2/3 of the corpus were used to train the SVM classifier and 1/3 of the corpus was used as a test set. This division yielded a training corpus of roughly 8,500 n-grams and a test corpus of around 4,500 n-grams. Table 2 shows the range of parameter settings we have tried to achieve optimal performance on the SVM. We achieved the best performance using Radial Basis Function (RBF) kernels for which the kernel degree and kernel coefficient values have no influence on the performance. Setting the cost value $C = 1$ and $bias = 1$, as well, yielded the best results. Classifying the test set with just the SVM yielded precision of 78.1%, recall of 82.08% for opinionated n-grams and an F1 value ($F1 = 2 * \frac{precision*recall}{precision+recall}$) of 80.04 (see table 3). This constituted the baseline which the system was compared to. The classification results using the system as described above yielded precision of 83.7%, recall of 84.49% for opinionated n-grams and an F1 value of 84.09 (see table 3). The arguments needed to classify n-grams using A-SVM were provided manually. Using A-SVM thus yielded a performance increase of 5.6% in precision, 2.41% in recall and 4.05 points for the F1 measure.

	Precision	Recall	F1
SVM	78.1%	82.08%	80.04
A-SVM	83.7%	84.49%	84.09

Table 3: Evaluation results of SVM and A-SVM

5.2 Qualitative Analysis

Both the system and the SVM evaluation rely in their workings on qualitative judgements that have either been passed during the development of the corpus or during the classification process itself. Additionally, analysing text with regards to its sentiment often involves ambiguities that are owed not just to the context words and phrases are set in, but also the context a pieces of text may be written or read in or who it is written or read by. For this reason, complementing a quantitative evaluation such as the one described in the previous section with a qualitative evaluation has allowed us to gain a clearer understanding of whether or not the system is performing in a suitable manner. The qualitative evaluation was conducted with nine users and worked as follows: Each user was asked to use the system twice, once analysing a piece of text that was provided and once choosing his or her own text to be analysed. Having multiple users judge the same piece of text meant attaining easily comparable results while having the users choose their own text allowed an analysis of the system's robustness to unexpected input. After each of the two system executions the user was asked to judge a number of statements on a fivefold scale indicating how much the user agreed with the statement made. Figure 3 shows an excerpt from the questionnaire.

In addition to passing judgement on statements such as those shown in figure 3, the user was also asked to judge the general system's quality on a scale from one to five. This was asked to complement the more specific questions with a broader evaluation of the user's confidence in the system's performance and output. The average scores given to the system by this measure were 3.8 ($std = 0.414$) for the classification of the text that was given and at 3.667 ($std = 0.408$) for the user's own input. The remaining questions were aimed at judging both the perceived ease of use of the system and the perceived value of providing user feedback. For these questions we achieved scores similar to the overall classification scores.

6. RELATED WORK

Argumentation has, to the best of our knowledge, not been used to this date within the setting of Sentiment Analysis. It has however been applied in unison with Machine Learning techniques in other settings, e.g. [22, 23]. While we have focused on analysing any generic text input, many researchers have tended to focus their efforts on the analysis of customer reviews, e.g. [24, 28]. Most solutions, such as our own, have either been directed at extracting opinionated contents from text, e.g. [3, 7], or at identifying opinionated contents as being either positive or negative, e.g. [13, 30, 34]. Only some have proposed holistic systems that encompass the complete analysis of text, for example [33].

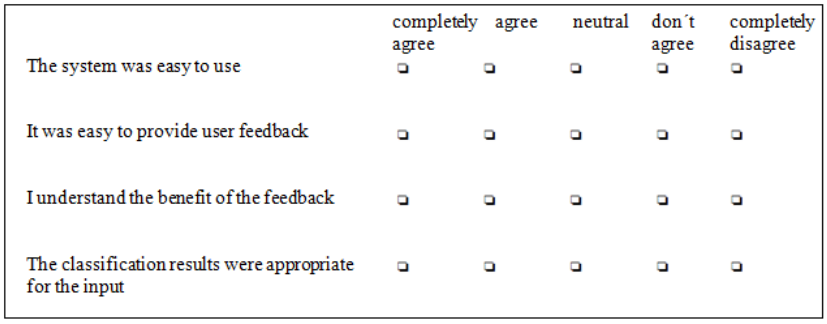
With rising interest in Sentiment Analysis, a number of text corpora, tailored to the needs and demands of Sentiment Analysis, have been developed. The most widely used have been the Multi-Perspective Question Answering (MPQA) corpus described in [32], which we have incorporated into our corpus, and the TREC (Text REtrieval Conference) blog tracks [19, 27]. The MPQA corpus annotates news articles and the TREC blog tracks use blog entries as source text.

[8] groups existing research into four categories: Keyword spotting, Lexical affinity, Statistical methods and Concept-based approaches. Lexical affinity assigns probabilistic values to words that determine how *affine* those words are to either other words [9] or an emotion. Our use of SentiWordNet scores as features of n-grams is similar to this concept. Concept-based approaches offer a deeper analysis of text, focusing on the semantics of it [2, 26]. This is realised through the use of web ontologies or semantic networks [6, 15]. We integrate some conceptual knowledge in our system via the user feedback.

Statistical methods have by far received the most attention in the Sentiment Analysis community and numerous algorithms have been used to approach the issue. Though supervised learning techniques (SVM as well as others) have been among the more popular ones researchers have also proposed applications using both Unsupervised Learning techniques, e.g. [3, 10, 34], and, more recently, Reinforcement Learning techniques, e.g. [7, 11, 33]. In [16] Kim and Hovy present a system for determining sentiment polarity which uses the concept of seed words to construct a classification model. A small amount of such seed words is collected which are either unambiguously positive or negative and annotated accordingly. This list is then iteratively expanded using synonymy and antonymy relations in WordNet. Such an approach requires significantly less effort than manually constructing a text corpus. Since we are classifying n-grams rather than words, however, a seed word approach was not feasible. In [7] Breck and colleagues use Conditional Random Fields (CRF) to distinguish opinionated text from non opinionated content based on the MPQA corpus. They also collect a number of features describing the text, among which are syntactical features determined by a POS tagging system, and have a CRF algorithm subsequently classify expressions according to those features. Choi and colleagues [11] apply CRF not to identify opinions but rather to find sources of opinions. Using various features that determine syntactic, semantic, and orthographic lexical characteristics of text, they train a CRF to identify both sources of direct and indirect opinions.

7. CONCLUSION

We have described A-SVM, a novel multimodal system for discriminating opinionated text from non-opinionated text that combines standard SVM and arguments from user feedback. We have trained and evaluated our system on a novel corpus and have shown experimentally that A-SVM outperforms standard SVM on the given corpus. We have additionally conducted a small qualitative analysis of A-SVM, the results of which show a rather favourable judgement of the users who have tried the system and, despite the relatively small sample size of nine users, we are confident that these results reflect the performance of A-SVM rather well. The results corroborate our hypothesis in section 1 that combining substantially different methods of of dealing with written language computationally should allow an increase of



	completely agree	agree	neutral	don´t agree	completely disagree
The system was easy to use	❑	❑	❑	❑	❑
It was easy to provide user feedback	❑	❑	❑	❑	❑
I understand the benefit of the feedback	❑	❑	❑	❑	❑
The classification results were appropriate for the input	❑	❑	❑	❑	❑

Figure 3: Excerpt from the questionnaire filled out by the evaluation participants

general performance compared to applying each method in isolation. Though we have both trained and evaluated A-SVM on a specific collection of news articles (as our corpus is based on the MPQA corpus [32]) during the qualitative evaluation the users were asked to submit text of their choice to A-SVM. The quality of the resulting classifications hints that with some effort we may be able to achieve a certain degree of domain independence. Though we have focused on using SVM for our classification, future efforts should be directed towards evaluating different algorithms, such as Maximum Entropy classifiers. In addition to testing different algorithms we need to conduct more extensive evaluations for each of them, including cross validation and more varied parameter settings.

The main issue left unaddressed in our work is the determination of opinion polarity. Numerous approaches have been presented to address this issue, e.g. [10, 30, 34]. Some of these efforts have focused on sentiment polarity exclusively, while others cover the full spectrum from determining opinionatedness to the summarisation of classification results. In light of the proposed system, there are two basic ways how polarity determination may be integrated into the system:

1. Add additional passes of binary decisions to the classification process
2. Develop a classifier that is able to make decisions on multiple classes at once

While the first alternative may prove to be the simpler solution, it may bring with it excessive computational demands and thus prove to be an unsustainable quick fix. In contrast to this solution, the second approach to integrating opinion polarity into the system's concerns would require fundamental changes to the system architecture but may prove beneficial with regards to computational efficiency. In addition to this rather central challenge, future improvements upon A-SVM may include tasks such as enhancing and extending the corpus, using different corpora to train the system and validate its performance or integrating further learning processes which continuously update a data base that contains not just the original corpus, but all past user inputs, as well.

By achieving measurable improvements upon a unimodal pattern recognition procedure we believe to have made a strong case for the potential benefits of going beyond pattern recognition algorithms in Sentiment Analysis. As has been suggested by some, e.g. Pat Langley in [17], it may prove to be necessary in the future to shift focus away from sheer statistical analysis to more complex tasks as envisioned when Machine Learning was still in its infancy. As Langley states:

> *"I do not believe that we should abandon any of the computational advances that have occurred in the [past] 25 years [...]. Each has been a valuable contribution to our understanding of learning. However, I think it is equally important that we not abandon the many insights revealed during the field's early period, which remain as valid today as when they initially came to light. The challenge for machine learning is to recover the discipline's original breadth of vision [...]."*

We argue that the concept of achieving text classification by combining established Machine Learning algorithms with Argumentation techniques allows us to make a step to achieving the above mentioned breadth of vision. Though subject to further investigation, this may hold true not just for basic binary classification of opinionatedness, but also multi-class classification for Sentiment Analysis as well as other NLP problems, such as Word Sense Disambiguation [25], Paraphrasing [1] or Argumentation Mining [21].

8. REFERENCES

[1] I. Androutsopoulos and P. Malakasiotis. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38(1):135–187, 2010.

[2] A. Balahur, J. M. Hermida, and A. Montoyo. Detecting implicit expressions of sentiment in text based on commonsense knowledge. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, pages 53–60. Association for Computational Linguistics, 2011.

[3] M. Baroni and S. Vegnaduzzo. Identifying subjective adjectives through web-based mutual information. In *Proceedings of KONVENS*, volume 4, pages 17–24. Citeseer, 2004.

[4] T. J. M. Bench-Capon and P. E. Dunne. Argumentation in artificial intelligence. *Artif. Intell.*, 171(10-15):619–641, 2007.

[5] C. Bishop. *Pattern recognition and machine learning*, volume 4. Springer New York, 2006.

[6] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.

[7] E. Breck, Y. Choi, and C. Cardie. Identifying expressions of opinion in context. In *Proceedings of the 20th international joint conference on Artifical intelligence*, pages 2683–2688. Morgan Kaufmann Publishers Inc., 2007.

[8] E. Cambria, B. Schuller, Y. Q. Xia, and C. Havasi. New avenues in opinion mining and sentiment analysis. *IEEE Intelligent Systems*, 28(2):15–21, 2013.

[9] D. Carmel, E. Farchi, Y. Petruschka, and A. Soffer. Automatic query refinement using lexical affinities with maximal information gain. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 283–290. ACM, 2002.

[10] Y. Choi and C. Cardie. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 793–801. Association for Computational Linguistics, 2008.

[11] Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 355–362. Association for Computational Linguistics, 2005.

[12] P. Clark and T. Niblett. The cn2 induction algorithm. *Machine learning*, 3(4):261–283, 1989.

[13] K. Dave, S. Lawrence, and D. Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th international conference on World Wide Web*, pages 519–528. ACM, 2003.

[14] A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422. Citeseer, 2006.

[15] M. Grassi, E. Cambria, A. Hussain, and F. Piazza. Sentic web: A new paradigm for managing social media affective information. *Cognitive Computation*, 3(3):480–489, 2011.

[16] S. Kim and E. Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, 2004.

[17] P. Langley. The changing science of machine learning. *Machine Learning*, 82:275–279, 2011.

[18] B. Liu. Sentiment analysis and subjectivity. *Handbook of Natural Language Processing*, 2, 2010.

[19] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the TREC 2007 blog track. In *Proceedings of TREC 2007*, 2007.

[20] G. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[21] R. Mochales and M. Moens. Argumentation mining. *Artificial Intelligence and Law*, 19(1):1–22, 2011.

[22] M. Možina, J. Žabkar, T. Bench-Capon, and I. Bratko. Argument based machine learning applied to law. *Artificial Intelligence and Law*, 13(1):53–73, 2005.

[23] M. Možina, J. Zabkar, and I. Bratko. Argument based machine learning. *Artificial Intelligence*, 171(10-15):922–937, 2007.

[24] T. Mullen and N. Collier. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of EMNLP*, volume 4, pages 412–418, 2004.

[25] R. Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10, 2009.

[26] D. J. Olsher. Full spectrum opinion mining: Integrating domain, syntactic and lexical knowledge. In *Data Mining Workshops (ICDMW), 2012 IEEE 12th International Conference on*, pages 693–700. IEEE, 2012.

[27] I. Ounis, M. De Rijke, C. Macdonald, G. Mishne, and I. Soboroff. Overview of the TREC-2006 blog track. In *Proceedings of TREC*, volume 6. Citeseer, 2006.

[28] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.

[29] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[30] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[31] H. Schmid. Probabilistic part-of-speech tagging using decision trees, 1994.

[32] J. Wiebe, T. Wilson, and C. Cardie. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2):165–210, 2005.

[33] T. Wilson, P. Hoffmann, S. Somasundaran, J. Kessler, J. Wiebe, Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. OpinionFinder: A system for subjectivity analysis. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, pages 34–35. Association for Computational Linguistics, 2005.

[34] H. Yu and V. Hatzivassiloglou. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 conference on Empirical methods in natural language processing-Volume 10*, pages 129–136. Association for Computational Linguistics, 2003.