Proceedings of the Seventeenth Conference on Computational
Natural Language Learning (CoNLL), Sofia, Bulgaria, August 2013, pp. 124--132.

# Frame Semantics for Stance Classification

**Kazi Saidul Hasan** and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{saidul,vince}@hlt.utdallas.edu

## Abstract

Determining the stance expressed by an author from a post written for a two-sided debate in an online debate forum is a relatively new problem in opinion mining. We extend a state-of-the-art learning-based approach to debate stance classification by (1) inducing lexico-syntactic patterns based on syntactic dependencies and semantic frames that aim to capture the meaning of a sentence and provide a generalized representation of it; and (2) improving the classification of a test post via a novel way of exploiting the information in other test posts with the same stance. Empirical results on four datasets demonstrate the effectiveness of our extensions.

## 1 Introduction

Given a post written for a *two-sided* topic in an online debate forum (e.g., *"Should abortion be allowed?"*), the task of *debate stance classification* involves determining which of the two sides (i.e., *for* or *against*) its author is taking. For example, a stance classification system should determine that the author of the following post is anti-abortion.

Post 1: *Abortion has been legal for decades and no one seems to have a problem with it. That's ridiculous! There are millions of people in the world who would love to have children but can't.*

Previous approaches to debate stance classification have focused on three debate settings, namely congressional floor debates (Thomas et al., 2006; Bansal et al., 2008; Balahur et al., 2009; Yessenalina et al., 2010; Burfoot et al., 2011), company-internal discussions (Murakami and Raymond, 2010), and online social, political, and ideological debates in public forums (Agrawal et al., 2003; Somasundaran and Wiebe, 2010; Wang and Rosé, 2010; Biran and Rambow, 2011; Hasan and Ng, 2012). As Walker et al. (2012) point out, debates in public forums differ from congressional debates and company-internal discussions in terms of language use. Specifically, online debaters use colorful and emotional language to express their points, which may involve sarcasm, insults, and questioning another debater's assumptions and evidence. These properties can potentially make stance classification of online debates more challenging than that of the other two types of debates.

Our goal in this paper is to improve the state of the art in stance classification of online debates, focusing in particular on *ideological debates*. Specifically, we present two extensions, one linguistic and the other extra-linguistic, to the state-of-the-art supervised learning approach to this task proposed by Anand et al. (2011). In our linguistic extension, we induce patterns from each sentence in the training set using *syntactic dependencies* and *semantic frames* that aim to capture the *meaning* of a sentence and provide a *generalized representation* of it. Note that while Anand et al.'s lexico-syntactic approach aims to generalize from a sentence using syntactic dependencies, we aim to generalize using semantic frames. As we will see in Section 4, not only is there no guarantee that syntactic dependencies can retain or sufficiently capture the meaning of a sentence during the generalization process, it is in fact harder to generalize from syntactic dependencies than from semantic frames. In our extra-linguistic extension, we improve the classification of a test post via a novel way of exploiting the information in other test posts with the same stance.

We evaluate our approach to stance classification of ideological debates on datasets collected for four domains from online debate forums. Experimental results demonstrate the effectiveness of our approach: it outperforms an improved version of Anand et al.'s approach by 2.6–7.0 accuracy points on the four domains.

| Domain | Number of posts | % of "for" posts |
|---|---|---|
| ABO | 1741 | 54.9 |
| GAY | 1376 | 63.4 |
| OBA | 985 | 53.9 |
| MAR | 626 | 69.5 |

Table 1: Statistics of the four datasets.

The rest of the paper is organized as follows. We first present our datasets in Section 2. Section 3 describes our two learning-based baseline systems for stance classification. Sections 4 and 5 discuss our two extensions. Finally, we show evaluation results in Section 6 and present conclusions in Section 7.

## 2 Datasets

For our experiments, we collect debate posts from four popular *domains*, Abortion (ABO), Gay Rights (GAY), Obama (OBA), and Marijuana (MAR). Each post should receive one of two *domain labels*, *for* or *against*, depending on whether the author of the post *supports* or *opposes* abortion, gay rights, Obama, or the legalization of marijuana. To see how we obtain these domain labels, let us first describe the data collection process in more detail.

We collect our debate posts for the four domains from an online debate forum[1]. In each domain, there are several two-sided debates. Each debate has a subject (e.g., "Abortion should be banned") for which a number of posts were written by different authors. Each post is manually tagged with its author's stance (i.e., *yes* or *no*) on the debate subject. Since the label of each post represents the subject stance but not the domain stance, we need to automatically convert the former to the latter. For example, for the subject "Abortion should be banned", the subject stance *yes* implies that the author opposes abortion, and hence the domain label for the corresponding label should be *against*.

We construct one dataset for each domain. Statistics of these datasets are shown in Table 1.

## 3 Baseline Systems

We employ as baselines two stance classification systems, Anand et al.'s (2011) approach and an enhanced version of it, as described below.

Our first baseline, Anand et al.'s approach, is a supervised method that trains a stance classifier for determining whether the stance expressed in a debate post is *for* or *against*. Hence, we create one training instance from each post in the training set, using the stance it expresses as its class label. Following Anand et al., we represent a training instance using five types of features: $n$-grams, document statistics, punctuations, syntactic dependencies, and, if applicable, the set of features computed for the immediately preceding post in its thread. Their $n$-gram features include both the unigrams and bigrams in a post, as well as its first unigram, first bigram, and first trigram. The features based on document statistics include the post length, the number of words per sentence, the percentage of words with more than six letters, and the percentage of words as pronouns and sentiment words. The punctuation features are composed of the repeated punctuation symbols in a post. The dependency-based features have three variants. In the first variant, the pair of arguments involved in each dependency relation extracted by a dependency parser is used as a feature. The second variant is the same as the first except that the head (i.e., the first argument in a relation) is replaced by its part-of-speech (POS) tag. The features in the third variant, the *topic-opinion features*, are created by replacing each feature from the first two types that contains a sentiment word with the corresponding polarity label (i.e., $+$ or $-$). For instance, given the sentence "John hates guns", the topic-opinion features $John^-$ and $guns^-$ are generated, since "hate" has a negative polarity and it is connected to "John" and "guns" via the *nsubj* and *dobj* relations, respectively. In our implementation, we train the stance classifier using SVM$^{light}$ (Joachims, 1999). After training, we can apply the stance classifier to classify the test instances, which are generated in the same way as the training instances.

Related work on stance classification of *congressional debates* has found that enforcing *author constraints* (ACs) can improve classification performance (e.g., Thomas et al. (2006), Burfoot et al. (2011), Lu et al. (2012)). ACs are a type of inter-post constraints that specify that two posts written by the same author for the same debate domain should have the same stance. We hypothesize that ACs could similarly be used to improve stance classification of ideological debates, and therefore propose a second baseline where we enhance the first baseline with ACs. Enforcing ACs is simple.

[1]http://www.createdebate.com/

We first use the learned stance classifier to classify the test posts as in the first baseline, and then *post-process* the labels of the test posts. Specifically, we sum up the confidence values[2] assigned to the set of test posts written by the same author for the same debate domain. If the sum is positive, then we label *all* the posts in this set as *for*; otherwise we label them as *against*.

## 4 Semantic Generalization

Our first extension to Anand et al.'s (2011) approach involves semantic generalization.

To motivate this extension, let us take a closer look at Anand et al.'s attempt to generalize using syntactic dependencies. Note that any approach that aims to generalize using syntactic dependencies suffers from several weaknesses. First, the semantic relationship between the pair of lexical items involved in each of these features is not encoded. This means that the resulting features do not adequately capture the meaning of the underlying sentence. Second, replacing a word with its POS tag is a syntactic, not semantic, generalization, and doing so further abstracts the resulting feature from the meaning of the underlying sentence. Above all, while the resulting features are intended to improve generalizations, they can provide very limited generalizations. To see why, consider two semantically similar sentences "I hate arrogant people" and "I dislike arrogant people". Ideally, any features that intend to provide a generalized representation of these sentences should be able to encode the fact that they are semantically similar. However, Anand et al.'s features would fail to do so because they cannot capture the fact that "hate" and "dislike" are semantically similar.

In the rest of this section we describe how we generate a *semantic generalization* of a sentence to capture its *meaning*. Our approach to semantic generalization involves (1) inducing from the training data a set of patterns that aim to provide a semantic generalization of the sentences in the training posts and (2) using them in combination with the baseline systems to classify a test post. Below we describe these two steps in detail.

### 4.1 Step 1: Pattern Induction

This step is composed of two sub-steps.

[2]We use as the confidence value the signed distance of the associated test point from the SVM hyperplane.

#### 4.1.1 Sub-step 1: Topic Extraction

For each domain, we extract a list of *topics*. We define a topic as a word sequence that (1) starts with zero or more adjectives and ends with one or more nouns and (2) appears in at least five posts from the domain. Using this method, for example, we can extract "abortion", "partial-birth abortion", "birth control", etc., as the topics for Abortion.

#### 4.1.2 Sub-step 2: Pattern Creation

Given a sentence, we create patterns to capture its information using syntactic dependencies and semantic frames.[3] These patterns can be divided into three types, as described below. For ease of exposition, we will use the two (semantically equivalent) sentences below as our running examples and see what patterns are created from them.

(1) Some people hate guns.
(2) Some people do not like guns.

**Subject-Frame-Object (SFO) patterns.** We create a set of SFO patterns for a transitive verb if (1) it is a frame target[4]; (2) its subject (respectively object) is a topic; and (3) its object (respectively subject) is a frame target. In sentence (1), *hate* is the target of the frame *Experiencer_focus* (henceforth EF), its subject, *people*, is a topic, and its object, *guns* is the target of the frame *Weapon*. As a result, we create a set of SFO patterns, each of which is represented as a 6-tuple. More specifically, we create the 8 SFO patterns shown in the first column of Table 2. Pattern 1 says that (1) this is an SFO pattern; (2) the subject is the word *people*; (3) the frame name of the verb is EF; (4) the frame name of the object is Weapon; (5) the verb is *not* negated (POS); and (6) we *don't care* (DC) whether the verb is sentiment-bearing. If the verb is sentiment-bearing (in this case, *hate* has a negative sentiment), we create another pattern that is the same as the first one, except that DC is replaced with its sentiment value (see Pattern 2).

Next, note that since the subject of *hate* is the target of the frame *People* and its object is a topic, we need to create patterns in a similar manner, resulting in Patterns 3 and 4. Note that *People* in these two patterns (with 'P' capitalized) is the

[3]We use the Stanford parser (de Marneffe and Manning, 2008) and SEMAFOR (Das et al., 2010) to obtain dependency relations and semantic frames, respectively.

[4]A word $w$ is the target of a frame $f$ if $f$ is assigned to $w$ to generalize its meaning. For example, *assassination*, *kill*, and *terminate* are the targets of the frame *Killing*.

| | | |
|---|---|---|
| 1 <SFO:people:EF:Weapon:POS:DC> | 9 <SFO:people:EF:Weapon:NEG:DC> | 17 <DF:dobj:EF:Weapon:POS:DC> |
| 2 <SFO:people:EF:Weapon:POS:−> | 10 <SFO:people:EF:Weapon:POS:−> | 18 <DF:dobj:EF:Weapon:POS:−> |
| 3 <SFO:People:EF:guns:POS:DC> | 11 <SFO:People:EF:guns:NEG:DC> | 19 <DF:dobj:EF:guns:POS:DC> |
| 4 <SFO:People:EF:guns:POS:−> | 12 <SFO:People:EF:guns:POS:−> | 20 <DF:dobj:EF:guns:POS:−> |
| 5 <SFO:people:EF:DC:POS:DC> | 13 <SFO:people:EF:DC:NEG:DC> | 21 <FET:people:Experiencer:EF:POS:DC> |
| 6 <SFO:people:EF:DC:POS:−> | 14 <SFO:people:EF:DC:POS:−> | 22 <FET:people:Experiencer:EF:POS:−> |
| 7 <SFO:DC:EF:guns:POS:DC> | 15 <SFO:DC:EF:guns:NEG:DC> | 23 <FET:guns:Content:EF:POS:DC> |
| 8 <SFO:DC:EF:guns:POS:−> | 16 <SFO:DC:EF:guns:POS:−> | 24 <FET:guns:Content:EF:POS:−> |

Table 2: Sample patterns created for sentences (1) and (2).

name of the frame *People*, not the word *people* appearing in the sentence.

To provide better generalization, we create a simplified version of each SFO pattern by replacing the frame name representing subject/object with the value DC. This results in Patterns 5–8.

For sentence (2), we can generate patterns in a similar manner, resulting in Patterns 9–16. For example, Pattern 9 contains the element NEG, which encodes the fact that the verb *like* is negated. Pattern 10 deserves discussion. Since the positive sentiment-bearing verb *like* is negated, the sentiment value of Pattern 10 is $-$, which encodes the fact that *not like* has a negative sentiment. The negation value of Pattern 10 is POS rather than NEG, reflecting the fact that *not like* does not appear in a negative context. In other words, the sentiment value needs to be flipped if the verb is negated, and so may the negation value. It is worth noting that Patterns 2 and 10 are identical, which provides suggestive evidence that sentences (1) and (2) are semantically equivalent.

**Dependency-Frame (DF) patterns.** We create a set of DF patterns for a dependency relation $d$ if (1) both arguments of $d$ are frame targets or (2) the head is a frame target and the dependent is a topic. For example, in the dependency relation *dobj*(*hate*,*guns*), both *hate* and *guns* are frame targets, as discussed above, and *guns* is a topic, so a set of DF patterns (Patterns 17–20 in Table 2) will be created from it. A DF pattern is represented as a 6-tuple. For example, Pattern 17 says that (1) this is a DF pattern; (2) the relation type is *dobj*; (3) the frame name of the head is EF; (4) the frame name of the dependent is *Weapon*; (5) the head is not negated; and (6) we don't care about the sentiment of the head. Pattern 18 is the same as Pattern 17, except that it takes into account the sentiment value of the verb. Patterns 19 and 20 replaces the frame name of the dependent with the topic name, which is *guns*. The negation and sentiment values are computed in the same way as those in the SFO patterns.

**Frame-Element-Topic (FET) patterns.** We create one FET pattern for every ($v$,$fe$) pair in a sentence where $v$ is a verb *and* a frame target, and $fe$ is a topic *and* a frame element of $v$'s frame.[5] In sentence (1), *people* is a topic and it is assigned the role *Experiencer*, so two FET patterns (Patterns 21 and 22) are created. Also, since *guns* is a topic and it is assigned the role *Content*, two additional FET patterns (Patterns 23 and 24) are created. The negation and sentiment values are computed in the same way as those in the SFO patterns.

### 4.2 Step 2: Classification

In this step, we will use the patterns learned in Step 1 in combination with the baseline systems to classify a test post. A simple way to combine the learned patterns with the baseline systems would be to augment the feature set they employ with the learned patterns. One potential weakness of this method is that the impact of these patterns could be undermined by the fact that they are significantly outnumbered by the baseline features, particularly the n-gram features.

For this reason, we decided to train another stance classifier, which we will refer to as the semantics-based classifier, $c_s$. Like the baseline stance classifier $c_b$, (1) $c_s$ is trained using SVM$^{light}$, (2) each training instance for $c_s$ corresponds to a training post, and (3) its class label is the stance the post expresses. Unlike $c_b$, however, the features employed by $c_s$ are created from the learned patterns. Specifically, from each pattern we create one binary feature whose value is 1 if and only if the corresponding pattern is applicable to the training post under consideration.

A natural question, then, is: how can we combine the decisions made by $c_b$ and $c_s$? To answer this question, we applied both classifiers to the de-

[5]Note that since $fe$ is a frame element of $v$'s frame, it is assigned a semantic role.

| System | ABO | GAY | OBA | MAR |
|---|---|---|---|---|
| $c_b$ | 60.3 | 63.2 | 59.5 | 67.1 |
| $c_s$ | 56.1 | 58.7 | 56.0 | 65.2 |

Table 3: Development set accuracies.

| System | ABO | GAY | OBA | MAR |
|---|---|---|---|---|
| $c_b$ | 22.9 | 18.5 | 24.1 | 9.6 |
| $c_s$ | 17.6 | 14.3 | 19.4 | 7.2 |

Table 4: Percentage of posts predicted correctly by one but not both classifiers on the development set.

velopment set for each domain and obtained the results in Table 3. As we can see, $c_s$ performs significantly worse than $c_b$ for all domains.[6]

At first glance, we should just abandon $c_s$ because of its consistently poorer performance. However, since the two classifiers are trained on disjoint feature sets (one is lexico-syntactic and the other semantic), we hypothesize that the mistakes they made on the development set could be *complementary*. To confirm this hypothesis, we compute the percentage of posts in the development set that are correctly classified by one but not the other. Results of this experiment are shown in Table 4. As we can see, these results are largely consistent with our hypothesis. For instance, for ABO, 22.9% of the posts are classified correctly only by $c_b$ but not $c_s$, whereas 17.6% of them are classified correctly only by $c_s$ but not $c_b$.

Given these results, we hypothesize that performance could be improved by combining the predictions made by $c_b$ and $c_s$. Since $c_b$ consistently outperforms $c_s$ on all datasets, we use $c_s$ to make a prediction if and only if (1) $c_b$ cannot predict confidently and (2) $c_s$ can predict confidently. This preference for $c_b$ is encoded in the following rule-based strategy for classifying a test post $p$, where the rules are applied in the order in which they are listed.

**Rule 1:** if $c_b$ can classify $p$ confidently, then use $c_b$'s prediction.

**Rule 2:** if $c_s$ can classify $p$ confidently, use $c_s$'s prediction.

**Rule 3:** use $c_b$'s prediction.

The next question is: how do we define *confidence*? Since $c_b$ and $c_s$ are SVM-based classifiers, the data points that are closer to the hyperplane are those whose labels the SVM is less confident about. Hence, we define confidence for classifier $c_i$ by the interval [$conf_l^i$, $conf_u^i$], where $conf_l^i < 0$ and $conf_u^i > 0$ are signed distances from the hyperplane defining $c_i$. Specifically, we say that a point $p$ is confidently classified by $c_i$ if and only if $p$ lies outside the interval defined by $conf_l^i$ and $conf_u^i$. Since we have two classifiers, $c_b$ and $c_s$, we need to define two intervals (i.e., four numbers). Rather than defining these four numbers by hand, we tune them jointly so that the accuracy of our combination strategy on the development set is maximized.[7]

There is a caveat, however. Recall that when applying this extension, we need to compute the signed distances of every post $p$ from $c_b$ and $c_s$ to determine which classifier will be used to classify $p$. The question, then, is: when applying this extension to the second baseline (the Anand et al. baseline extended with ACs) where all the posts written by the same author for the same domain should have the same stance, how should their signed distances be computed? We adopt a simple solution: we take the average of the signed distances of all such posts from the corresponding hyperplane and set the signed distance of each such post to the average value.

## 5 Exploiting Same-Stance Posts

To classify a debate post $p$ in the test set, we have so far exploited only the information extracted from $p$ itself. However, it is conceivable that we can improve the classification of $p$ by exploiting the information extracted from other test posts that have the same stance as $p$. This is the goal of our second extension.

To see why doing so can improve the classification of $p$, we make a simple observation: some posts are easier to classify than the others. Typically, posts containing expressions that are strong indicators of the stance label are easier to classify than those that do not. As an example, consider the following posts:

Post 2: *I don't think abortion should be illegal.*

Post 3: *What will you do if a woman's life is in danger while she's pregnant? Do you still want to sacrifice her life simply because the fetus is alive?*

It should be fairly easy for a human to see that the authors of both posts support abortion. However, Post 2 is arguably easier to classify than

[6]All significance tests are paired $t$-tests, with $p < 0.05$.

[7]For parameter tuning, for each of the four numbers we tried the values from $-0.5$ to $+0.5$ with a step value of 0.001.

Post 3: Post 2 has an easy-to-determine stance, whereas Post 3 has a couple of rhetorical questions that may be difficult for a machine to understand. Hence, we might be able to improve the classification of Post 3 by exploiting information from other posts that have the same stance as itself (which in this case would be Post 2).

In practice, however, we are not given the information of which posts have the same stance. In the two subsections below, we discuss two simple methods of determining whether two posts are *likely* to have the same stance.

### 5.1 Using Same-Author Information

The first method, which we will refer to as $M_1$, is fairly straightforward: we posit that two posts are likely to have the same stance if they are written by the same author. Given a test post $p$ to be classified, we can use this method to identify a subset of $p$'s same-stance posts. For convenience, we denote this set as SameStancePosts($p$). The question, then, is: how can we exploit information in SameStancePosts($p$) to improve the classification of $p$? One way would be to *combine* the content of the posts in SameStancePosts($p$) with that of $p$ (i.e., by taking the union of all the binary-valued feature vectors), and use the class value of the combined post as the class value of $p$. However, rather than simply combining all the posts to form one big post, we generalize this idea by (1) generating all possible combinations of posts in SameStancePosts($p$); (2) for each such combination, combine it with $p$; (3) classify each combination obtained in (2) using the SVM classifier; (4) sum the confidence values of all the combinations; and (5) use the signed value as the class value of $p$. Note that if SameStancePosts($p$) contains $n$ posts, the number of possible combinations is $\sum_{i=0}^{n} \binom{n}{i}$. For efficiency reasons, we allow each combination to contain at most 10 posts.

At first glance, it seems that the combination method described in the previous paragraph is an alternative implementation of ACs. (Recall that ACs are inter-post constraints that ensure that two posts written by the same author for the same domain should receive the same label.) Nevertheless, there are two major differences between our combination method and ACs. First, in ACs, the same-author posts can only interact via the confidence values assigned to them. On the other hand, in our proposal, the same-author posts interact via feature sharing. In other words, in ACs, the same-author posts interact *after* they are classified by the stance classifier, whereas in our proposal, the interaction occurs *before* the posts are classified. Second, in ACs, all the same-author posts receive the same stance label. On the other hand, this is not necessarily the case in our proposal, because two same-author posts can be classified using different combinations. In other words, ACs and our combination method are not the same. In fact, they can be used in conjunction with each other.

| Feature | Definition |
|---|---|
| SameDebate | whether authors posted in same debate |
| SameThread | whether authors posted in same thread |
| Replied | whether one author replied to the other |

Table 5: Interaction features for the author-agreement classifier.

### 5.2 Finding Similar-Minded Authors

Using $M_1$ to identify same-stance posts has a potential weakness. If an author has composed a small number of posts, then the number of combinations that can be generated will be small. In the extreme case, if an author has composed just one post $p$, then no combinations will be generated using $M_1$.

To enable $p$ to benefit from our idea of exploiting same-stance posts, we propose another method to identify same-stance posts, $M_2$, which is a generalization of $M_1$. In $M_2$, we posit that two posts are likely to have the same stance if they are written by the same author or by *similar-minded* authors. Given test post $p$, we can compute SameStancePosts($p$) using the definition of $M_2$, and apply the same 5-step combination method described in the previous subsection to SameStancePosts($p$) to classify $p$.

The remaining question is: given an author, $a$, in the test set, how do we compute his set of similar-minded authors, $A_{similar}$? To do this, we train a binary author-agreement classifier on the training set to generate $A_{similar}$ for $a$. Specifically, each training instance corresponds to a pair of authors in the training set having one of two class labels, *agree* (i.e., authors have the same stance) and *disagree* (i.e., authors have opposing stances). We represent each instance with two types of features. Features of the first type are obtained by taking the *difference* of the feature vectors corresponding to the two authors under consideration, where the feature vector of an author is obtained by taking the union of the feature vectors corresponding to all of the posts written by her. Taking the difference would allow the learner to focus on those features whose values differ in the feature vectors. For the second type of features, we use *author interaction* information encoded as three binary features (see Table 5 for their definitions), which capture how authors interact with each other in a debate thread. After training the classifier, we apply it to classify the author-pairs in the test set. Then, for each author $a$, we compute her $k$-nearest authors based on the magnitude of their agreement, where $k$ is tuned to maximize accuracy on the development data.[8] Finally, we take $A_{similar}$ to be the set of $k$-nearest authors.

## 6 Evaluation

### 6.1 Experimental Setup

Results are expressed in terms of *accuracy* obtained via 5-fold cross validation, where accuracy is the percentage of test instances correctly classified. Since all experiments require the use of development data for parameter tuning, we use three folds for model training, one fold for development, and one fold for testing in each fold experiment.

### 6.2 Results

Results are shown in Table 6. Row 1 shows the results of the Anand et al. (2011) baseline on the four datasets, obtained by training a stance classifier using the SVM$^{light}$ package.[9] Row 2 shows the results of the second baseline, Anand et al.'s system enhanced with ACs. As we can see, incorporating ACs into Anand et al.'s system improves its performance significantly on all datasets and yields a system that achieves an average improvement of 4.6 accuracy points.

Next, we incorporate our first extension, pattern induction, into the better of the two baselines (i.e., the second baseline). Results of combining $c_b$ and $c_s$ to classify the test posts (together with the ACs) are shown in row 3 of Table 6. As we can see, incorporating pattern induction into the second baseline significantly improves its performance on all four datasets and yields a system that achieves an average improvement of 2.48 accuracy points.

Before incorporating our second extension, let us recall our earlier hypothesis that combining $c_b$ and $c_s$ using our method would be better than training just one classifier that combines the features used by $c_b$ and $c_s$. The reason behind our hypothesis was that simply combining the feature sets would undermine the impact of pattern-based features because they would be significantly outnumbered by the features in $c_b$. To confirm this hypothesis, we showed in row 4 of Table 6 the results of this experiment, where we trained one classifier on all the features used by $c_b$ and $c_s$. As we can see, this classifier (referred to as $c_{bs}$ in the table) together with the ACs performs significantly worse than the $c_b$+$c_s$+AC system (row 3) on all datasets. In fact, the $c_b$+AC system (row 2) outperforms the $c_{bs}$+AC system on OBA, but they are statistically indistinguishable on the remaining datasets. These results suggest that combining the pattern-based features with the baseline features into one feature set renders the former ineffective.

| System | ABO | GAY | OBA | MAR |
|---|---|---|---|---|
| $c_b$ | 61.4 | 62.6 | 58.1 | 66.9 |
| $c_b$+AC | 72.0 | 64.9 | 62.7 | 67.8 |
| $c_b$+$c_s$+AC | 73.2 | 68.0 | 64.2 | 71.9 |
| $c_{bs}$+AC | 71.8 | 65.0 | 60.2 | 67.9 |
| $c_b$+$c_s$+$M_1$+AC | 74.8 | 69.1 | 69.7 | 73.2 |
| $c_b$+$c_s$+$M_2$+AC | **75.9** | **70.6** | **71.2** | **75.3** |

Table 6: 5-fold cross-validation accuracies.

Finally, we incorporate our second extension, the one that involves generating combinations of test posts written by the same author ($M_1$) and by both the same author and similar-minded authors ($M_2$). Results of these experiments are shown in rows 5–6 of Table 6. The $M_1$-based system significantly outperforms $c_b$+$c_s$+AC on all four domains, yielding an average improvement of 2.4 accuracy points. The $M_2$-based system further beats the $M_1$-based system by 1.5 accuracy points on average, and their performance difference is significant on all but the ABO domain.

Overall, our two extensions yield a stance classification system that significantly outperforms the better baseline on all four datasets, with an average improvement of 6.4 accuracy points.

Given the better performance of the combination-based systems, a natural question is: can we further improve performance by applying our combination methods to generate artificial posts and use them as additional training instances? To answer this question, we apply both $M_1$ and $M_2$ to generate additional

[8] We tested values of $k$ from 1 to 7.

[9] For all SVM experiments, the regularization parameter C is tuned using development data, but the remaining learning parameters are set to their default values.

training instances, using a random selection of same-stance authors in place of $M_2$'s k-nearest neighbor method. However, neither method yields an improvement in performance over the method on which it is based. We speculate that since all the posts in the training combinations are already present in the training set as individual posts, they are more likely to be farther away from the hyperplane than the individual posts, meaning that they are less likely to be support vectors. This in turn implies that they are less likely to affect classification performance.

### 6.3 Error Analysis

To gain additional insights into our approach, we performed a qualitative analysis of the errors produced by our best-performing system below.

**Failure to accumulate decisions from several clues.** Authors often express their stance using a group of sentences where the latter sentence(s) indicate the actual stance and the initial sentence(s) may give a false impression about the author's stance. Consider Post 1 (see Section 1) and Post 4.

Post 4: *I agree abortion creates stress and pain. I agree it kills a potential life. That does not mean it is right to ban abortion.*

In Post 1, the author is anti-abortion, whereas in Post 4, the author is pro-abortion. However, the first sentence in Post 1 gives a misleading clue about the author's stance, and so do the first two sentences in Post 4. Since all the systems discussed in the paper operate on one sentence at a time, they are all prone to such errors. One way to address this problem could be to determine how adjacent sentences are related to each other via the use of discourse relations.

**Presence of materials irrelevant to stance.** Because of the informal style of writing, we often find long posts with one or two sentences indicating the actual stance of the author. The rest of such posts often include descriptions of an author's personal experience, comments or questions directed to other authors etc. Such long posts are frequently misclassified for all four domains. Consider the following example.

Post 5: *Marijuana should at least be decriminalized. Driving stoned, however, is something totally different and should definitely be a crime. Also, weed can't kill you, unlike cigarettes and alcohol. In my opinion cigarettes should definitely be illegal, but they're so ingrained into our culture that I doubt that is going to happen any time soon.*

In this post, the author supports the legalization of marijuana. However, the only useful hints about her stance are "marijuana should at least be decriminalized" and "weed can't kill you". The rest of the post is not helpful for stance classification.

**Convoluted posts appearing later in long post sequences.** As a post sequence gets longer, authors tend to focus on specific aspects of a debate and consequently, it becomes more difficult to classify their stances, even with the context-based features (features taken from the immediately preceding post) proposed by Anand et al. Consider the following post sequence, where only the first post (P1) and the nth post (Pn) are shown due to space limitations.

**[P1: Anti-Obama]** Obama is a pro-abortionist. Killing babies is wrong so stop doing it. The new health reform bill is not good. There are some good things but more worse than good. You could have just passed some laws instead of making a whole bill.

...

**[Pn: Pro-Obama]** Killing fetuses isn't wrong. Besides, we could use those fetuses for stem cell research.

As we can see, the author of P1 does not support Obama because of his pro-abortion views. In Pn, a pro-Obama author explains why she thinks abortion is not wrong. However, without the context from P1 that Obama is pro-abortion, it is not easy for a machine to classify Pn correctly. This problem is more serious in ABO and GAY than in the other domains as the average length of a post sequence in these two domains is larger.

## 7 Conclusions

We examined the under-studied task of stance classification of ideological debates. Employing our two extensions yields a system that outperforms an improved version of Anand et al.'s approach by 2.6–7.0 accuracy points. In particular, while existing approaches to debate stance classification have primarily employed lexico-syntactic features, to our knowledge this is the first attempt to employ FrameNet for this task to induce features that aim to capture the meaning and provide semantic generalizations of a sentence. In addition, our method for identifying and exploiting same-stance posts during the inference procedure provides further gains when used on top of our FrameNet extension.

## References


Rakesh Agrawal, Sridhar Rajagopalan, Ramakrishnan Srikant, and Yirong Xu. 2003. Mining newsgroups using networks arising from social behavior. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 529–535.

Pranav Anand, Marilyn Walker, Rob Abbott, Jean E. Fox Tree, Robeson Bowmani, and Michael Minor. 2011. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA 2011)*, pages 1–9.

Alexandra Balahur, Zornitsa Kozareva, and Andrés Montoyo. 2009. Determining the polarity and source of opinions expressed in political debates. In *Proceedings of the 10th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '09, pages 468–480.

Mohit Bansal, Claire Cardie, and Lillian Lee. 2008. The power of negative thinking: Exploiting label disagreement in the min-cut classification framework. In *Proceedings of the 22nd International Conference on Computational Linguistics: Companion volume: Posters*, pages 15–18.

Or Biran and Owen Rambow. 2011. Identifying justifications in written dialogs. In *Proceedings of the 2011 IEEE Fifth International Conference on Semantic Computing*, ICSC '11, pages 162–168.

Clinton Burfoot, Steven Bird, and Timothy Baldwin. 2011. Collective classification of congressional floor-debate transcripts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1506–1515.

Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 948–956.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Proceedings of the COLING Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, CrossParser '08, pages 1–8.

Kazi Saidul Hasan and Vincent Ng. 2012. Predicting stance in ideological debate with rich linguistic knowledge. In *Proceedings of the 24th International Conference on Computational Linguistics: Posters*, pages 451–460.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In Bernhard Scholkopf and Alexander Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 44–56. MIT Press.

Yue Lu, Hongning Wang, ChengXiang Zhai, and Dan Roth. 2012. Unsupervised discovery of opposing opinion networks from forum discussions. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 1642–1646.

Akiko Murakami and Rudy Raymond. 2010. Support or oppose? Classifying positions in online debates from reply activities and opinion expressions. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 869–875.

Swapna Somasundaran and Janyce Wiebe. 2010. Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, CAAGET '10, pages 116–124.

Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 327–335.

Marilyn Walker, Pranav Anand, Rob Abbott, and Ricky Grant. 2012. Stance classification using dialogic properties of persuasion. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 592–596.

Yi-Chia Wang and Carolyn P. Rosé. 2010. Making conversational structure explicit: Identification of initiation-response pairs within online discussions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 673–676.

Ainur Yessenalina, Yisong Yue, and Claire Cardie. 2010. Multi-level structured models for document-level sentiment classification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1046–1056.