In this paper, we present work on extracting social networks from unstructured text. We introduce novel features derived from semantic annotations based on FrameNet. We also introduce novel semantic tree kernels that help us improve the performance of the best reported system on social event detection and classification by a statistically significant margin. We show results for combining the models for the two aforementioned sub-tasks into the overall task of social network extraction. We show that a combination of features from all three levels of abstractions (lexical, syntactic and semantic) are required to achieve the best performing system.

1 Introduction

Social network extraction from text has recently been gaining a considerable amount of attention (Agarwal and Rambow, 2010; Elson et al., 2010; Agarwal et al., 2013a; Agarwal et al., 2013b; He et al., 2013). One of the reason for this attention, we believe, is that being able to extract social networks from unstructured text may provide a powerful new tool for historians, political scientists, scholars of literature, and journalists to analyze large collections of texts around entities and their interactions. The tool would allow researchers to quickly extract networks and assess their size, nature, and cohesiveness, a task that would otherwise be impossible with corpora numbering millions of documents. It would also make it possible to make falsifiable claims about these networks, bringing the experimental method to disciplines like history, where it is still relatively rare.
sub-tasks, but no evaluation was presented for the task of social network extraction. We experiment with two different designs of combining models for the two sub-tasks: 1) One-versus-All and 2) Hierarchical. We find that the hierarchical design outperforms the more commonly used One-versus-All by a statistically significant margin.

Following are the contributions of this paper:

1. We design and propose novel frame semantic features and tree-based representations and show that tree kernels are well suited to work with noisy semantic parses.

2. We show that in order to achieve the best performing system, we need to include features and tree structures from all levels of abstractions, lexical, syntactic, and semantic, and that the convolution kernel framework is well-suited for creating such a combination.

3. We combine the previously proposed sub-tasks (social event detection and classification) into a single task, social network extraction, and show that combining the models using a hierarchical design is significantly better than the one-versus-all design.

The rest of the paper is structured as follows: In Section 2, we give a precise definition of the task and describe the data. In Section 3, we give a brief overview of frame semantics and motivate the need to use frame semantics for the tasks addressed in this paper. In Section 4, we present semantic features and tree kernel representations designed for the tasks. In Section 5, we briefly review tree kernels and support vector machines (SVM). In Section 6 we present experiments and discuss the results. In Section 7 we discuss related work. We conclude and give future directions of work in Section 8.

2 Data and Task Definition

In Agarwal et al. (2010), we presented the annotation details of social events on a well-known corpus – Automated Content Extraction\(^1\) (ACE2005). We defined a social event to be a happening between two entities (of type person) \(E_1\) and \(E_2\) \((E_1 \neq E_2)\), in which at least one entity is cognitively aware of the other and of the happening taking place. We defined two broad categories of social events: Interaction (INR) and Observation (OBS). In a social event of type INR, the two participating entities are mutually aware of each other, i.e., INR is a bi-directional social event. For example, meetings and dinners are social events of type interaction. In a social event of type OBS, only one of the two participating entities is aware of the other and therefore, OBS is a one-directional social event, directed from the entity that is aware of the other to the other entity. For example, thinking about someone, or missing someone are social events of type OBS. Table 1 shows the distribution of the data. There are 199 INR type of social events, 199 OBS events, and 1,609 pairs of entity mentions have no event between them.

Task definition: The task is, given a pair of entity mentions in a sentence, to predict if the entities are participating in a social event or not (social event detection, SED), and if they are, to further predict the type of social event (INR or OBS, social event classification, SEC). In this paper, we evaluate our system on the above tasks as well as a combined task: social network extraction (SNE): given a sentence and a pair of entity mentions, predict the class of the example from one of the following three categories: {No-Event, INR, OBS}.

For the purposes of this paper, we use gold named entity mentions to avoid errors caused due to named entity recognition systems. This is a common practice used in the literature for reporting relation extraction systems (Zelenko et al., 2003; Kambhatla, 2004; Zhao and Grishman, 2005; GuoDong et al., 2005; Harabagiu et al., 2005; Nguyen et al., 2009). We use standard terminology from the literature to refer to the pair of entities mentions as target entities \(T_1\) and \(T_2\).

3 Frame Semantics and FrameNet

FrameNet (Baker et al., 1998) is a resource which associates words of English with their meaning. Word meanings are based on the notion of “semantic frame”. A frame is a conceptual description of a type of event, relation, or entity, and it

<table>
<thead>
<tr>
<th></th>
<th>No-Event</th>
<th>INR</th>
<th>OBS</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Examples</td>
<td>1,609</td>
<td>199</td>
<td>199</td>
</tr>
</tbody>
</table>

Table 1: Data distribution; INR are interaction social events. OBS are observation social events.
includes a list of possible participants in terms of the roles they play; these participants are called “frame elements”. Through the following example, we present the terminology and acronyms that will be used throughout the paper.

Example (2) shows the frame annotations for the sentence *Toujan Faisal said she was informed of the refusal by an Interior Ministry committee*. One of the semantic frames in the sentence is **Statement**. The frame evoking element (FEE) for this frame is *said*. It has two frame elements (FE): one of type **Speaker** (*Toujan Faisal*) and the other of type **Message** (*she was informed ... by an Interior Ministry committee*).

(2) \( \text{[FE-Speaker} \text{Toujan Faisal]} \text{]} \text{[FEE-Statement} \text{said]} \text{]} \text{[FE-Message} \text{she was informed of the refusal by an Interior Ministry committee]} \)

In example (2), the speaker of the message (*Toujan Faisal*) is mentioning another group of people (the *Interior Ministry committee*) in her message. By definition, this is a social event of type OBS. In general, there is an OBS social event between any **Speaker** and any person mentioned in the frame element **Message** of the frame **Statement**. This close relation between frames and social events is the reason for our investigation and use of frame semantics for the tasks addressed in this paper.

### 4 Feature space and data representation

We convert examples\(^2\) into two kinds of structured representations: feature vectors and tree structures. Each of these structural representations may broadly be categorized into one or more of the following levels of abstraction: {Lexical, Syntactic, Semantic}. Table 2 presents this distribution. Our final results show that all of our top performing models use a data representation that is a combination of features and structures from all levels of abstraction. We review previously proposed features and tree structures in subsections 4.1, 4.2, and 4.3. To the best of our knowledge, the remaining features and structures presented in this section are novel.

#### 4.1 Bag of words (BOW)

We create a vocabulary from our training data by using the Stanford tokenizer (Klein and Manning, 2003) followed by removal of stop words and Porter Stemming. We convert each example \((\vec{x})\) to a set of three boolean vectors: \(\{b_1, b_2, b_3\}\). \(b_1\) is the occurrence of words before the first target, \(b_2\) between the two targets and \(b_3\) after the second target. Here the first target and second target are defined in terms of the surface order of words. Though these features have been previously proposed for relation extraction on ACE (GuoDong et al., 2005), they have not been utilized for the task we address in this paper.

#### 4.2 Syntactic structures (AR2010)

In Agarwal and Rambow (2010), we explored a wide range of syntactic structures for the two tasks of social event detection (SED) and classification (SEC). All our previous structures were derived from a variation of two underlying tree structures: phrase structure trees and dependency trees. The best structure we proposed was PET_GR_SqGRW, which was a linear combination of two tree kernels and one word kernel: 1) a structure derived from a phrase structure tree (PET); 2) a grammatical role tree (GR), which is a dependency tree in which words are replaced with their grammatical roles; and 3) a path from one entity to the other in a dependency tree, in which grammatical roles of words are inserted as additional nodes between the dependent and parent (SqGRW). We refer the reader to Agarwal and Rambow (2010) for details of these structures. For the rest of the paper, we refer to this structure, PET_GR_SqGRW, as “AR2010”. We use AR2010 as one of our baselines.

#### 4.3 Bag of frames (BOF)

We use Semafor (Chen et al., 2010) for obtaining the semantic parse of a sentence. Semafor found instances of 1,174 different FrameNet frames in our corpus. Each example \((\vec{x})\) is converted to a vector of dimension 1,174, in which \(x_i\) (the \(i\)th component of vector \(\vec{x}\)) is 1 if the frame number \(i\) appears in the example, and 0 otherwise.

#### 4.4 Hand-crafted semantic features (RULES)

We use the manual of the FrameNet resource to hand-craft 199 rules that are intended to detect the presence and determine the type of social events between two entities mentioned in a sentence. An example of one such rule is given in section 3, which we reformulate here. We also present another example:
(3) If the frame is **Statement**, and the first target entity mention is contained in the FE **Speaker**, and the second is contained in the FE **Message**, then there is an OBS social event from the first entity to the second.

(4) If the frame is **Commerce_buy**, and one target entity mention is contained in the FE **Buyer**, and the other is contained in the FE **Seller**, then there is an INR social event between the two entities.

Each rule corresponds to a binary feature: it takes a value 1 if the rule fires for an input example, and 0 otherwise. Consider the following sentence:

(5) \[\text{[Coleman]}_{T1-Ind} \text{[the defendants]}_{T2-Grp} \]

\[\text{[he]}_{T1-Ind} \text{[bought]}_{T2-Grp} \text{[drugs]} \]

In this sentence, there are two social events: 1) an OBS event triggered by the word *claimed* between Coleman and defendants and 2) an INR event triggered by the word *bought* between he (co-referential with Coleman) and the defendants.

Semafor correctly detects two frames in this sentence: 1) the frame **Statement**, with Coleman as **Speaker**, and he bought ..., and 2) the frame **Commerce_buy**, with he as the **Buyer**, drugs as the **Goods** and the defendants as the **Seller**. Both hand-crafted rules (3 and 4) fire and the corresponding feature values for these rules is set to 1. Firing of these rules (and thus the effectiveness these features) is of course highly dependent on the fact that Semafor provides an accurate frame parse for the sentence.

**4.5 Semantic trees (FrameForest, FrameTree, FrameTreeProp)**

Semafor labels *text spans* in sentences as frame evoking elements (FEE) or frame elements (FE). A sentence usually has multiple frames and the frame annotations may overlap. There may be two ways in which spans overlap (Figure 1): (a) one frame annotation is completely embedded in the other frame annotation and (b) some of the frame elements overlap (in terms of text spans). We now present the three frame semantic tree kernel representations that handle these overlapping issues, along with providing a meaningful semantic kernel representation for the tasks addressed in this paper.

For each of the following representations, we assume that for each sentence \(s\), we have the set of semantic frames, \(F_s = \{F = \langle FEE, \{FE_1, FE_2, \ldots, FE_n]\rangle \}\) with each frame \(F\) having an FEE and a list of FEs. We illustrate the structures using sentence (5).

**4.5.1 FrameForest Tree Representation**

We first create a tree for each frame annotation \(F\) in the sentence. Consider a frame, \(F = \langle FEE, \{FE_1, FE_2, \ldots, FE_n]\rangle\). For the purposes of tree construction, we treat \(FEE\) as another \(FE\) (call it \(FE_0\)) of type **Target**. For each \(FE_i\), we choose the subtree from the dependency parse tree that is the smallest subtree containing all words annotated as \(FE_i\) by Semafor. Call this subtree extracted from the dependency parse \(DepTree_{FE_i}\). We then create a larger tree by adding \(DepTree_{FE_i}\) as a child of a new node labeled with frame element \(FE_i\): \((FE_i DepTree_{FE_i})\). Call this resulting tree \(SubTree_{FE_i}\). We then connect all the \(SubTree_{FE_i}\) \((i \in \{0, 1, 2, \ldots, n\})\) to a new root node labeled with the frame \(F: (F SubTree_{FE_0} \ldots SubTree_{FE_n})\). This is the tree for a frame \(F\). Since the sentence could have multiple frames, we connect the forest of frame trees to a new node called **ROOT**.
We prune away all subtrees that do not contain the target entities. We refer to the resulting tree as FrameForest.

For example, in Figure 2, the left tree is the FrameForest tree for sentence (5). There are two frames in this sentence that appear in the final tree because both these frames contain the target entities and thus are not pruned away. The two frames are Commerce_buy and Statement. We first create trees for each of the frames. For the Commerce_buy frame, there are three frame elements: Target (the frame evoking element), Buyer and Seller. For each frame element, we get the subtree from the dependency tree that contains all the words belonging to that frame element. The subtree for FEE Target is \((\text{bought } T1-\text{Ind} \text{ (from } T2-\text{Grp}))\). The subtree for FE Buyer is \((T1-\text{Ind})\) and the subtree for FE Seller is \((\text{from } T2-\text{Grp})\). We connect these subtrees to their respective frame elements and connect the resulting subtrees to the frame (Commerce_buy). Similarly, we create a tree for the frame Statement. Finally, we connect all frame trees to the ROOT.

In this representation, we have avoided the frame overlapping issues by repeating the common subtrees: the subtree \((\text{bought } T1-\text{Ind} \text{ (from } T2-\text{Grp}))\) is repeated under the FEE Target of the Statement frame as well as under the FE Message of the Statement frame.

### 4.5.2 FrameTree Representation

For the design of this tree, we deal with the two overlapping conditions shown in Figure 1 differently. If one frame is fully embedded in another frame, we add the former as a child of the latter frame. In Figure 2, the frame Commerce_buy is fully embedded in the frame element Message of the frame Statement. Therefore, the frame subtree for Commerce_buy appears as a subtree of Message.

If the frames overlap partially, we copy over the overlapping portions of the structures to each of the frame sub-trees.

For the design of this representation, we remove all lexical nodes (struck out nodes in Figure 2) and trees that do not span any of the target entities (not shown in the figure). As a result, this structure is the smallest semantic structure that contains the two target entities. The right tree in Figure 2 is the FrameTree tree for sentence (5).

#### 4.5.3 FrameTreeProp Tree Representation

We are using a partial tree kernel (PTK) for calculating the similarity of two trees (as detailed in section 5). The PTK does not skip over nodes of the tree that lie on the same path. For establishing an OBS social event between Coleman and the defendants, all the structure needs to encode is the fact that one target appears as a Speaker and the other appears in the Message (of the speaker). In FrameTree, this information is encoded but in an unclear manner – there are two nodes (Commerce_buy and Seller) that come in between the node Message and T2-Grp.

For this reason, we copy the nodes labeled with the target annotations \((T1 \sim *, T2 \sim *)\) to all nodes (that are frame elements of a frame) on the path from them to the root in FrameTree. We call this
variation of FrameTree, in which we propagate \( T1 - * \), \( T2 - * \) nodes to the root, FrameTreeProp. For the running example, FrameTreeProp will be: \( \text{(Statement (Speaker T1-Ind) (Message (Commerce_buy ...)) (T2-Grp)))} \). Using this tree representation, one of the sub-trees in the implicit feature space will be \( \text{(Statement (Speaker T1-Ind) (Message (T2-Grp)))} \), which encodes the relation between the two targets in a more direct manner as compared to FrameTree.

5 Machine Learning

We represent our data in form of feature vectors and tree structures. We use convolution kernels (Haussler, 1999) that make use of the dual form of Support Vector Machines (SVMs). In the dual form, the optimization problem that SVM solves is the following (Burges, 1998):

\[
\max \sum_{i} \mu_{i} - \sum_{i,j} \mu_{i} \mu_{j} y_{i} y_{j} K(x_{i}, x_{j})
\]

\[\text{s.t. } \sum_{i} \mu_{i} y_{i} = 0 \]

\[\mu_{i} \geq 0 \forall i = 1, 2, \ldots, l\]

Here, \( x_{i} \) is the input example, \( y_{i} \) is the class of the example \( x_{i} \), \( \mu_{i} \) is the Lagrange multiplier associated with example \( x_{i} \), \( l \) is the number of training examples, and \( K \) is the kernel function that returns a similarity between two examples. More formally, \( K \) is the function, \( K : X \times X \to \mathbb{R} \), that maps a pair of objects belonging to the set \( X \) to a real number. For example, if we represent our input examples as feature vectors, the set \( X \) would be the set of feature vectors. For feature vectors, we use a linear kernel, i.e. \( K(x_{i}, x_{j}) = x_{i} \cdot x_{j} \) (dot product of the two vectors). For our tree representations, we use a Partial Tree Kernel (PTK), first proposed by Moschitti (2006). PTK is a relaxed version of the Subset Tree (SST) kernel proposed by Collins and Duffy (2002). A subset tree kernel measures the similarity between two trees by counting all subtrees common to the two trees. However, there is one constraint: all daughter nodes of a parent node must be included (in the sub-trees). In PTK, this constraint is removed. Therefore, in contrast to SST, PT kernels compare many more substructures. For a combination of feature vectors and tree representations, we simply use the linear combination of their respective kernels.

6 Experiments and Results

We present 5-fold cross-validation results on the ACE2005 corpus annotated for social events. Since the number of types of features and structures is not large (Table 2), we run an exhaustive set of \( 2^{7} - 1 = 127 \) experiments for each of three tasks: Social Event Detection (SED), Social Event Classification (SEC) and Social Network Extraction (SNE). To avoid over-fitting to a particular partition into folds, we run each 5-fold experiment 50 times, for 50 randomly generated partitions. The results reported in the following tables are all averaged over these 50 partitions. The absolute standard deviation on an average is less than 0.004. This means that the performance of our models across 50 random folds does not fluctuate and hence the system is robust. We use McNemar’s significance test and refer to statistical significance as \( p < 0.05 \).

6.1 Social event detection (SED) and classification (SEC)

We report precision (P), recall (R) and F1 measure for the detection task, and % accuracy for the classification task. For both these tasks, our previous best performing system was PET_GR_SqGRW (which we refer to as AR2010). We use this as a baseline, and introduce two new baselines: the bag-of-words (BOW) baseline and a linear combination of BOW and AR2010, referred to as BOW_AR2010.

Table 3 presents the results for these two tasks for various features and structures. The results show that our purely semantic models (RULES, BOF, FrameTree, FrameTreeProp) do not perform well alone. FrameForest, which encodes some lexical and syntactic level features (but is primarily semantic), also performs worse than the baselines when used alone. However, a combination of lexical, syntactic and semantic structures improves the performance by an absolute of 1.1% in F1-measure for SED (from 0.574 to 0.585). This gain is statistically significant. For SEC, the absolute gain from our best baseline (BOW_AR2010) is 0.8% in F1-measure (from 82.3 to 83.1), which is not statistically significant. However, the gain of 2% from our previously proposed best system (AR2010) is statistically significant.
Table 3: Results for three tasks: “SED” is Social Event Detection, “SEC” is Social Event Classification, “SNE” is Social Network Extraction. The first three models are the baseline models. The next five models are the novel semantic features and structures we propose in this paper. “All” refers to the model that uses all the listed structures together. “BOW_AR2010_FrameForest_FrameTreeProp” refers to the model that uses a linear combination of mentioned structures. AR2010_FrameTreeProp is a linear combination of AR2010 and FrameTreeProp.

6.2 Social Network Extraction (SNE)

Social network extraction is a multi-way classification task, in which, given an example, we classify it into one of three categories: {No-Event, INR, OBS}. A popular technique of performing multi-way classification using a binary classifier like SVM, is one-versus-all (OVA). We try this along with a less commonly used technique, in which we stack two binary classifiers in a hierarchy. For the hierarchical design, we train two models: (1) the SED model (INR + OBS versus No-Event) and (2) the SEC model (INR versus OBS). Given a test example, it is first classified using the SED model. If the prediction is less than zero, we label it as No-Event. Otherwise, the test example is passed onto SEC and finally classified into either INR or OBS.

We see that none of the semantic features and structures alone outperform the baseline. However, a combination of structures from different levels of abstraction achieve the best performance: an absolute gain of 1.5% in F1 (statistically significant) when we use a hierarchical design (from 0.465 to 0.480).

Comparing hierarchical versus OVA approaches, we observe that the hierarchical approach outperforms the OVA approach for all our models by a statistically significant margin. The performance for our best reported model (AR2010_FrameTreeProp) for OVA in terms precision, recall, and F1-measure is 0.375, 0.592, 0.459 respectively. This is statistically significantly worse than hierarchical approach (0.397, 0.608, 0.480).

6.3 Discussion of results

Performing well on SED is more important than SEC, because if a social event is not detected in the first place, the goodness of the SEC model is irrelevant. Therefore, the best feature and structure combination we report in this paper is a combination of AR2010 and FrameTreeProp.

To gain insight into the how each type of semantic feature and structure contribute to our previously proposed lexical and syntactic model (AR2010), we perform experiments in which we add one semantic feature/structure at a time to AR2010. Table 4 presents the results for this study. We see that the hand-crafted RULES do not help in the overall task. We investigated the reason for RULES not being as helpful as we had expected. We found that when there is no social event, the rules fire in 7% of the cases. When there is a social event, they fire in 17% of cases. So while they fire more often when there is a social event, the percentage of cases in which they fire is small. We hypothesize that this is due the dependence of RULES on the correctness of se-
mantic parses. For example, Rule (4) correctly detects the social event in sentence (5), since Semafor correctly parses the input. In contrast, Semafor does not correctly parse the input sentence (1): it correctly identifies the Statement frame and its Message frame element, but it fails to find the Speaker. As a result, Rule (3) does not fire, even though the semantic structure is partially identified. This, we believe, highlights the main strength of tree kernels – they are able to learn semantic patterns, without requiring correctness or completeness of the semantic parse.

Out of the semantic structures we propose, FrameTreeProp adds the most value to the baseline system as compared to other semantic features and structures. This supports our intuition that we need to reduce unbounded semantic dependencies between the target entities by propagating the target entity tags to the top of the semantic tree.

<table>
<thead>
<tr>
<th>Model</th>
<th>SED (F1)</th>
<th>SEC (%A)</th>
<th>SNE Hier. (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR2010</td>
<td>0.574</td>
<td>81.1</td>
<td>0.465</td>
</tr>
<tr>
<td>+ RULES</td>
<td>0.576</td>
<td>80.8</td>
<td>0.465</td>
</tr>
<tr>
<td>+ BOF</td>
<td>0.569</td>
<td>80.7</td>
<td>0.459</td>
</tr>
<tr>
<td>+ FrameForest</td>
<td>0.571</td>
<td>82.6</td>
<td>0.472</td>
</tr>
<tr>
<td>+ FrameTree</td>
<td>0.579</td>
<td>81.5</td>
<td>0.473</td>
</tr>
<tr>
<td>+ FrameTreeProp</td>
<td>0.585</td>
<td>82.0</td>
<td>0.480</td>
</tr>
</tbody>
</table>

Table 4: A study to show which semantic features and structures add the most value to the baseline. The top row gives the performance of the baseline. Each consecutive row shows the result of the baseline plus the feature/structure mentioned in that row.

7 Related Work

There have been recent efforts to extract networks from text (Elson et al., 2010; He et al., 2013). However, these efforts extract a different type of network: a network of only bi-directional links, where the links are triggered by quotation marks. For example, Elson et al. (2010) and He et al. (2013) will extract an interaction link between Emma and Harriet in the following sentence. However, their system will not detect any interaction links in the other examples mentioned in this paper.

(6) “Take it,” said Emma, smiling, and pushing the paper towards Harriet “it is for you. Take your own.”

Our approach to extract and classify social events builds on our previous work (Agarwal and Rambow, 2010), which in turn builds on work from the relation extraction community (Nguyen et al., 2009). Therefore, the task of relation extraction is most closely related to the tasks addressed in this paper. Researchers have used other notions of semantics in the literature such as latent semantic analysis (Plank and Moschitti, 2013) and relation-specific semantics (Zelenko et al., 2003; Culotta and Sorensen, 2004). To the best of our knowledge, there is only one work that uses frame semantics for relation extraction (Harabagiu et al., 2005). Harabagiu et al. (2005) propose a novel semantic kernel that incorporates frame parse information in the kernel computation that calculates similarity between two dependency trees. They, however, do not propose data representations that are based on frame parses and the resulting arborescent structures, instead adding features to syntactic trees. We believe the implicit feature space of kernels based on our data representation encode a richer and larger feature space than the one proposed by Harabagiu et al. (2005).

8 Conclusion and Future Work

This work has only scratched the surface of possibilities for using frame semantic features and tree structures for the task of social event extraction. We have shown that tree kernels are well suited to work with possibly inaccurate semantic parses in contrast to hand-crafted features that require the semantic parses to be completely accurate. We have also extended our previous work by designing and evaluating a full system for social network extraction.

A more natural data representation for semantic parses is a graph structure. We are actively exploring the design of semantic graph structures that may be brought to bear with the use of graph kernels (Vishwanathan et al., 2010).

Acknowledgments

We would like to thank CCLS’s IT heads, Hatim Diab and Manoj Pooleery, for providing the infrastructure support. This paper is based upon work supported in part by the DARPA DEFT Program. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.
References


