ASM Kernel: Graph Kernel using Approximate Subgraph Matching for Relation Extraction

Nagesh C. Panyam, Karin Verspoor, Trevor Cohn and Kotagiri Ramamohanarao
Department of Computing and Information Systems,
The University of Melbourne, Australia
npanyam@student.unimelb.edu.au
{karin.verspoor, t.cohn, kotagiri}@unimelb.edu.au

Abstract

Kernel methods have been widely studied in several natural language processing tasks such as relation extraction and sentence classification. In this work, we present a new graph kernel that is derived from a distance measure described in prior work as Approximate Subgraph Matching (ASM). The classical ASM distance, shown to be effective for event extraction, is not a valid kernel and was primarily designed to work with rule based systems. We modify this distance suitably to render it a valid kernel (ASM kernel) and enable its use in powerful learning algorithms such as Support Vector Machine (SVM).

We compare the ASM kernel with SVMs to the classical ASM with a rule based approach, for two relation extraction tasks and show an improved performance with the kernel based approach. Compared to other kernels such as the Subset tree kernel and the Partial tree kernel, ASM kernel outperforms in relation extraction tasks and is of comparable performance in a general sentence classification task. We describe the advantages of the ASM kernel such as its flexibility and ease of modification, which offers further directions for improvement.

1 Introduction

Many natural language processing tasks such as relation extraction or question classification are cast as supervised classification problems (Bunescu and Mooney, 2005), with the object to classify being an entity pair or a sentence. Traditional approaches have typically focussed on transforming the input into a feature vector which is then classified using learning algorithms such as decision trees or SVM. A primary limitation of this approach has been the manual effort required to construct a rich set of features that can yield a high performance classification. This effort is evident for the construction of features from the syntactic parse of the text, which is often represented as an ordered structure such as a tree or a graph. Linearizing a highly expressive structure such as a graph, by transforming it into a flat array of features is inherently harder. This problem of constructing explicit feature sets for complex objects is generally overcome by kernel methods for classification. Kernel methods allow for an implicit exploration of a vast high dimensional feature space and shift the focus from feature engineering to similarity score design. Importantly, such a kernel must be shown to be symmetric and positive semi-definite (Burges, 1998), to be valid for use with kernelized learning algorithms such as SVM. Deep learning based approches (Zeng et al., 2014; Xu et al., 2015) are other alternatives to eliminate the manual feature engineering efforts. However, in this work we are primarily focussed on kernel methods.

In NLP, kernel methods have been effectively used for relation extraction and sentence classification. Subset tree kernels (SSTK) and partial tree kernels (PTK) were developed to work with constituency parse trees and basic dependency parse trees. However, these kernels are not suitable for arbitrary graph structures such as the enhanced dependency parses (Manning et al., 2014). Secondly, tree kernels can only handle node labels and not edge labels. As a work around, these kernels require that the original dependency graphs be heuristically altered to translate edge labels into special nodes to create different syntactic representations such as the grammatical relation centered tree (Croce et al., 2011). These limitations were overcome with the Approximate Subgraph

Matching (ASM) (Liu et al., 2013), that was designed to be a flexible distance measure to handle arbitrary graphs with edge labels and edge directions. However, the classic ASM is not a valid kernel and therefore cannot be used with powerful learning algorithms like SVM. It was therefore used in a rule-based setting, where it was shown to be effective for event extraction (Kim et al., 2011).

1.1 Contributions

In this work, our primary contribution is a new graph kernel (ASM kernel), derived from the classical approximate subgraph matching distance, that:

- is flexible, working directly with graphs with cycles and edge labels.
- is a valid kernel for use with powerful learning algorithms like SVM.
- outperforms classical ASM distance with rule based method for relation extraction.
- outperforms tree kernels for relation extraction and is of comparable performance for a sentence classification task.

2 Methods

In this section, we first describe the classical ASM distance measure that was originally proposed in (Liu et al., 2013). We then discuss the modifications we introduce to transform this distance measure into a symmetric, L_2 norm in a valid feature space. This step allows us to enumerate the underlying feature space and to elucidate the mapping from a graph to a vector in a high dimensional feature space. We then define the ASM kernel as a dot product in this high dimensional space of well defined features. Besides establishing the validity of the kernel, the feature map clarifies the semantics of the kernel and helps design of interpretable models.

2.1 Classic ASM distance

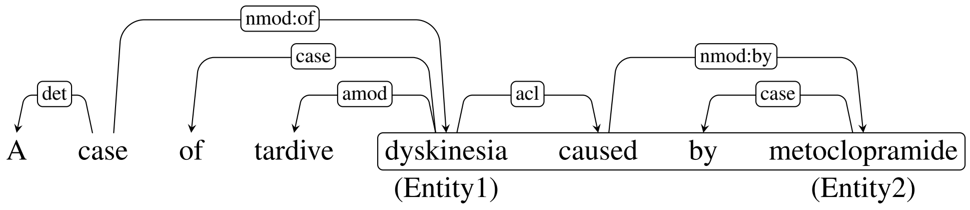
We describe the classic ASM distance in the context of a binary relation extraction task. Consider two sample sentences drawn from the training set and test set of such a task corpus, as illustrated in Figure 1. Entity annotations are given for the whole corpus, which are character spans referring to two entities in a sentence. In the illustrated example, the entities are chemicals (metoclopramide and pentobarbital) and diseases (dyskinesia and amnesia). The training data also contains relation annotations, which are related entity pairs (metoclopramide, dyskinesia). We assume that the relation (causation) is implied by the training sentence and then to try to infer a similar relation or its absence in the test sentence.

Preprocessing The first step in the ASM event extraction system is to transform each sentence to a graph, whose nodes represent tokens in the sentence. Node labels are derived from the corresponding tokens properties, such as the word lemma or part of speech (POS) tag or a combination of both. The node labels for entities are usually designated as *Entity1* and *Entity2*. This process is referred to as *entity blinding* and is known to improve generalization (Thomas et al., 2011). Labelled edges are given by a dependency parser (Manning et al., 2014). A graph from a test sentence is referred to as a *main graph*. Given a training sentence and its corresponding graph, we extract the subgraph within it, that consists of only those nodes that represent the entities or belong to the shortest path[1] between the two entities. This is referred to as a *rule subgraph* (see Figure 1a).

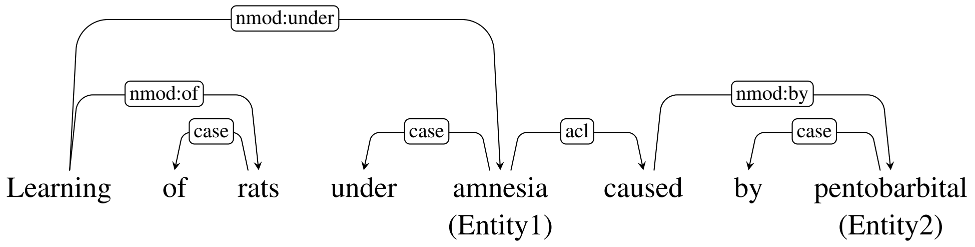
Approximate Subgraph Isomorphism The main idea in ASM is that a test sentence is considered to be of same type or express the same relation as that of a training sentence, if we can find a subgraph isomorphism of rule graph (training sentence) in the main graph (test sentence). Exact subgraph isomorphism (boolean) is considered too strict and is expected to hurt generalization. Instead, ASM tries to compute a measure (a real number) of subgraph isomorphism. This measure is referred to as the *Approximate Subgraph Matching* distance. If the ASM distance between a rule graph and main graph is within a predefined *threshold*, then the test sentence is considered positive, or of the same relation type as the rule graph.

ASM distance We first compute an injective mapping M from rule graph to main graph. An injective matching scheme essentially maps each node of the subgraph to a node in the main graph, with identical labels. If no matching scheme can be found, then the ASM distance is set to a very large quantity (∞) . Following the node matching, we do not demand a matching of edges between

[1]Throughout this paper, shortest path refers to the path with least number of edges in the undirected version of the graph.



(a) Graph from a training sentence. The rule subgraph within is shown with a surrounding box.



(b) Main graph from a test sentence.

Figure 1: Sample dependency graphs from two sentences expressing a relation of type "causation" between two entities.

the two graphs, like in a typical exact isomorphism search. Instead, we compute the difference between these edges to get an approximate subgraph matching (ASM) distance. The ASM distance is a weighted summation of 3 components, namely structural distance, label distance and directionality distance. These are described below, with the aid of notations described in Table 1. Note that edge directions are interpreted as special *directional labels* of type "forward" or "backward".

The structural distance (SD), label distance (LD) and the directionality distance (DD) for a path $P^r_{x,y}$ is defined as:

$$\begin{aligned} \text{SD}(P^r_{x,y}, P^m_{x',y'}) &= |\text{Len}(P^r_{x,y}) - \text{Len}(P^m_{x',y'})| \\ \text{LD}(P^r_{x,y}, P^m_{x',y'}) &= \#\text{EL}(P^r_{x,y}) \triangle \text{EL}(P^m_{x',y'}) \\ \text{DD}(P^r_{x,y}, P^m_{x',y'}) &= \#\text{DL}(P^r_{x,y}) \triangle \text{DL}(P^m_{x',y'}) \end{aligned} \tag{1}$$

Essentially, these distances reflect the differences in the structure of the two graphs, focussed at one vertex pair at a time. Notice that the above distances are defined for a given shortest path between two vertices. However, there could be multiple shortest paths of equal length between two vertices in a graph. The distances for a given vertex pair is taken to be the minimum distance over all possible choices for a shortest path.

If $SP(x, y)$ and $SP(x', y')$ denote the set of shortest paths for x, y in the rule graph and x', y' in the main graph respectively, the distance measures for the vertex pair x, y are defined as:

Symbol	Meaning
$G^m =< V^m, E^m >$	Main graph
$G^r =< V^r, E^r >$	Rule graph
$M = \{(x, x'), \ldots\}$, $\forall x \in V^r$ s.t , label(x) = label(x') and $x' \in V^m$.	M is the injective mapping scheme, i.e $M(x) = x'$
$P^g_{x,y}$	Shortest path in graph g between the vertices x, y.
Len(P)	The length of the path P.
EL(P)	The multiset of edge labels on the path P.
DL(P)	The multiset of directional labels on the path P.
$S_1 \triangle S_2$	The set symmetric difference of sets S_1 and S_2 .
$\|r_1 - r_2\|$	The absolute difference of real numbers r_1 and r_2
$\#S$	The cardinality of set S.

Table 1: Notations used in this paper

$$
\begin{aligned}
\text{SD}(x,y) &= \min_{\forall P \in SP(x,y), P' \in SP(x',y')} \text{SD}(P,P') \\
\text{LD}(x,y) &= \min_{\forall P \in SP(x,y), P' \in SP(x',y')} \text{LD}(P,P') \\
\text{DD}(x,y) &= \min_{\forall P \in SP(x,y), P' \in SP(x',y')} \text{DD}(P,P')
\end{aligned}
\tag{2}
$$

The (unnormalized) distance measure between two graphs G^r and G^m is obtained from the distances between all possible vertex pairs:

$$
\begin{aligned}
\text{SD}(G^r, G^m) &= \sum_{\forall x,y \in V^r} \text{SD}(x,y) \\
\text{LD}(G^r, G^m) &= \sum_{\forall x,y \in V^r} \text{LD}(x,y) \\
\text{DD}(G^r, G^m) &= \sum_{\forall x,y \in V^r} \text{DD}(x,y)
\end{aligned}
\tag{3}
$$

The final classical ASM distance in its unnormalized form is a weighted summation of the above 3 components:

$$
\begin{aligned}
\text{ASM}(G^r, G^m) =& w_1 \times \text{SD}(G^r, G^m) \\
&+ w_2 \times \text{LD}(G^r, G^m) \\
&+ w_3 \times \text{DD}(G^r, G^m)
\end{aligned}
\tag{4}
$$

The classic ASM distance includes a normalization process that we have not described, as it is not central to our discussion. Notice that this distance does not meet the conditions of a positive semi-definite kernel. Trivially, it is an asymmetric distance and undefined when an injective matching is unavailable. In the next section, we describe a modified form of ASM that is shown to be a $||L_2||$ norm in a valid feature space.

2.2 Modified ASM distance

Classical ASM distance measure evaluates distances between two pairs of vertices $x, y \in G^r$ and $x', y' \in G^m$ where $x' = M(x)$ and $y' = M(y)$ and M is the injective matching scheme. Recall that the simplest injective matching scheme maps vertices with identical labels. We assume that node labels in a graph are all distinct. Further, for all missing labels in a graph G, we (virtually) insert a single disconnected dummy node with that label. These steps ensure that all labels in the label vocabulary are represented in the graph G and map to unique vertices. We can now define the modified ASM distance over label pairs, instead of vertex pairs. Next, in the modified ASM distance, we consider only a single shortest path between a label pair x, y. For example, in Figure 1a the shortest path $P_{\text{“caused”},\text{“by”}} =$ (“caused”, “Entity2”, “by”). When more than one such shortest path is available (due to cycles), we simply choose the first of such paths as the representative shortest path. Finally, we transform the set of edge labels in each path into a vector representation (i.e, a bag of words representation). For example, $\text{EL}(P_{\text{“caused”},\text{“by”}}) =$ (“nmod:by” : 1, “case” : 1). We use the euclidean distance between these vectors, instead of the cardinality of set symmetric difference used in the classical ASM label distance. Directionality distance is modified similarly.

Symbol	Meaning
$P^g_{x,y}$	The representative shortest path in graph g between the vertices x, y.
$\text{EL}(P)$	The bag of words representation of edge labels on the path P.
$\text{DL}(P)$	The bag of words representation of directional labels on the path P.
$\|\|V_1 - V_2\|\|_2$	The $\|\|L_2\|\|$ norm of the vector difference $V_1 - V_2$
$L = \{a, an, the, \ldots\}$	The vocabulary of all node labels (lemmas)

Table 2: Additional notations for the modified ASM

The modified distances, namely structural distance (SD), label distance (LD) and directionality distance (DD) are defined below, following a few additional notations in Table 2:

$$\begin{aligned}
&\text{SD}(G^r, G^m) = \\
&\quad \sum_{\forall l_1, l_2 \in L \times L} (\text{Len}(P^r_{x,y}) - \text{Len}(P^m_{x',y'}))^2 \\
&\text{LD}(G^r, G^m) = \\
&\quad \sum_{\forall l_1, l_2 \in L \times L} \|\text{EL}(P^r_{x,y}) - \text{EL}(P^m_{x',y'})\|_2 \\
&\text{DD}(G^r, G^m) = \\
&\quad \sum_{\forall l_1, l_2 \in L \times L} \|\text{DL}(P^r_{x,y}) - \text{DL}(P^m_{x',y'})\|_2 \\
&\quad \text{where} \quad \text{label}(x) = \text{label}(x') = l_1 \\
&\quad \text{and} \quad \text{label}(y) = \text{label}(y') = l_2
\end{aligned} \tag{5}$$

The final modified ASM distance in its unnormalized form is a weighted summation of the above 3 components:

$$\begin{aligned}
\text{ASM}(G^r, G^m) =& w_1 \times \text{SD}(G^r, G^m) \\
&+ w_2 \times \text{LD}(G^r, G^m) \\
&+ w_3 \times \text{DD}(G^r, G^m)
\end{aligned} \tag{6}$$

2.3 ASM Kernel: Validity and Semantics

A valid kernel function is required to be symmetric and positive semidefinite (Burges, 1998). Also, from Mercer's condition (Burges, 1998), we note that such a kernel is essentially equivalent to a dot product ($K(x, y) = \sum_i \phi(x)_i \phi(y)_i$) in a euclidean (or a general RKHS) "feature" space and a valid mapping function(ϕ) exists, that transforms the input object to the feature space. Kernel validity can be directly established by deriving the underlying feature map (ϕ), and computing the kernel directly as a dot product of the mapped vectors ($\langle \phi(x), \phi(y) \rangle$). Also, this feature map directly relates to the semantics of the kernel and helps to interpret the resulting model. We follow this approach, to first derive the feature space ϕ underlying the modified ASM distance and show it to be equivalent to the $||L_2||$ norm in this feature space. That is, modified ASM distance$(x, y) = \sum_i (\phi(x)_i - \phi(y)_i)^2$. The ASM kernel can then be obtained by replacing the sum of squared differences with sum of products. That is, the ASM kernel$(x, y) = \sum_i \phi(x)_i \phi(y)_i$.

The feature space of structural distance can be indexed by the set $\{L \times L\}$ where L is the vocabulary of node labels. Each feature value is just the length of the shortest path between a label pair corresponding to the feature index. Similarly, the directionality distance corresponds to two sets of features indexed by $\{L \times L\}$. The first feature set counts the number of "forward" edges and the second set counts the number of "backward" edges in the shortest path corresponding to the feature index. Finally, the label distance can be seen as an extension of directionality distance, obtained by extending the set of edge labels from a limited set of two symbols {"forward", "backward"} to the finite set of all possible dependency edge labels.

Consider the example illustrated in Figure 1a. The ASM kernel transforms the rule graph in the above example, to an explicit set of features as described below. The set of node labels for this example is $L' = \{$"Entity1", "caused", "by", "Entity2"$\}$. The set of edge labels for this example is $EL' = \{$"acl", "nmod:by", "case"$\}$. The features corresponding to this example can be indexed by the set $L' \times \{EL' \cup \{$"distance", "forward", "backward"$\}\}$. Tuples such as ("Entity1", "caused", "distance") and ("Entity1", "by", "distance") represent structural distance features. The values for these are 1 and 3 respectively. Tuples such as ("Entity1", "caused", "forward") and ("Entity1", "by", "forward") denote the forward directionality features. The values for these are 1 and 2 respectively. Similarly, features for label distance are generated by considering the elements of EL'. For example, the edge label "acl" is associated with the tuples such as ("Entity1", "caused", "acl") and ("Entity1", "by", "acl"). The values for these features are 1 and 1 respectively. This is because, there is exactly one instance of the edge label "acl" in the shortest path from "Entity1" to "caused" and from "Entity1" to "by".

To summarize, we note that the modified ASM distance is a sum of squared differences of feature counts (real numbers) and the corresponding feature space is a finite and enumerable set as shown above. By replacing the sum of squares with sum of products of the feature counts, we obtain the dot product in the same valid feature space, which forms the ASM kernel.

2.4 Running times

The classical ASM distance has an exponential time complexity, typically brought about by the search for subgraph isomorphism. Still, it was shown to be practical, as text with very long sentences and therefore large dependency graphs are quite rare. However, the current ASM kernel evaluation is polynomial in runtime as there is no requirement for an isomorphism test.

Consider two graphs with m and n vertices and $m \leq n$ without loss of generality. We first identify the vertices with common labels across the graph, via a sort merge of label lists. This step takes $O(n \cdot \log(m))$. Next, note that there are at most m^2 label pairs that are common to two graphs. Each label pair corresponds to a path in the graph with at most n nodes. Enumerating the features of this graph involves a single traversal of each such path, which translates to a complexity bound of $O(n \cdot m^2)$ or simply $O(n^3)$ (a looser upper bound). Finding the shortest paths across all node pairs can be done in $O(n^3)$ time using standard graph algorithms (Seidel, 1995).

2.5 Implementation details

We used the Java based Kelp framework (Filice et al., 2015) for implementing and testing the ASM kernel with SVM. For this paper, the weights associated with the ASM method (w_1, w_2, w_3) are all set to 1. The feature set for each graph can be computed in parallel and cached. In practice, we found that SVM train-test cycle with ASM kernel took about a day for some of the largest datasets described in this paper. For classic ASM distance, we used the implementation[2] made freely available by the authors.

3 Evaluation

We described the classic ASM distance in the context of a binary relation extraction task in Section 2.1. In this section, we report the performance of this approach over two relation extraction tasks, namely the Chemical-induced-Disease (CID) task (Wei et al., 2015) and the Seedev-Binary relation extraction task (Chaix et al., 2016). Note that it is a rule based with a single parameter being the distance threshold. We determine the optimal threshold value with a grid search over the validation set. We compare this rule based system with a supervised classification approach for relation extraction. The dependency graph of a sentence is made entity aware, by setting the labels of the two nodes corresponding to entities as "Entity1" and "Entity2". Relation extraction is cast as the task of graph labelling using a multiclass SVM

[2]http://asmalgorithm.sourceforge.net

Kernel	P	R	F1
Classical ASM system	35.1	81.1	49.0
PTK with LCT	43.3	77.3	55.5
SSTK with CP	42.6	72.8	53.7
ASM Kernel with DP	46.4	77.7	**58.1**

Table 3: Results on CID test data for sentence level relations. Key: LCT= Location Centered Tree, CP = Constituency Parse, DP = Dependency Parse

with ASM kernel. We extend the comparison to two well known tree kernels, namely Subset Tree Kernel (SSTK) and the Partial Tree Kernel (PTK) that have been shown to be effective for relation extraction (Zelenko et al., 2002; Moschitti, 2006; Chowdhury et al., 2011). Note that unlike constituency parse trees, dependency trees have edge labels which cannot be handled by these tree kernels. Therefore, the edge labels are converted into node labels of specially inserted nodes in the original dependency graph, to get a modified structure referred to as the *Location Centered Tree* (LCT) (Lan et al., 2009).

Finally, we compare the ASM kernel with tree kernels in a sentence classification task. This is a straightforward application of kernels in a graph classification problem, over the unmodified dependency graphs of the corpus.

3.1 Chemical induced Disease (CID) task

The goal of the CID shared task (Wei et al., 2015) is to infer the "induces" relation between Chemicals and Disease from biomedical publications. The shared task has provided a corpus of 1500 PubMed abstracts, divided equally into training, development and test datasets. The corpus includes non-sentence relations, i.e. Chemical-Disease pairs whose relationship cannot be inferred by a single sentence and requires analyzing the whole abstract. We omit such relations and focus on extracting sentence level relations only. We compare ASM kernel against Subset tree kernels (SSTK) and Partial tree kernels (PTK) and report the results in Table 3.

3.2 Seedev Binary

The Seedev Binary task (Chaix et al., 2016) addresses the extraction of genetic and molecular mechanisms that regulate plant seed development from biomedical literature. The task organizers

Kernel	P	R	F1
Classical ASM	8.8	41.7	14.5
PTK with DP	11.6	38.0	17.8
SSTK with DP	14.4	24.6	18.2
SSTK with CP	14.8	32.6	20.3
ASM Kernel	25.0	27.9	**26.4**

Table 4: Aggregate results over the development dataset of the Seedev Binary task. Key:CP = Constituency Parse, DP = Dependency Parse

Kernel	Accuracy (% age)
Bag of Words	86.2
Partial Tree Kernel with LCT	90.6
Subset Tree Kernel with GRCT	**91.4**
ASM Kernel	89.6

Table 5: Results on Question Classification dataset.

provided paragraphs from manually selected full text publications on seed development of *Arabidopsis thaliana* annotated with mentions of biological entities like *proteins* and *genes*, and binary relations like *Exists_In_Genotype* and *Occurs_In_Genotype*. The Seedev task involves extraction of 22 different binary relations over 16 entity types. The corpus provided consists of a total of $7,082$ entities and $3,575$ binary relations, divided into training, development and test datasets. Entity mentions within a sentence and the events between them are provided in the gold standard annotations. We created 22 separate classifiers for detecting each of these relation types. A special feature of this task is that each relation type is associated with a type signature, that specifies which entity types are allowed for the given relation, as its two arguments. We use this type signature, to filter the corpus to create 22 separate training and test sets, for each of the 22 classifiers. In Table 4 aggregate results over the 22 event types are reported for the development set (the annotations for the test set are not available).

3.3 Question Classification

This task deals with classifying general questions expressed in natural language, into one of 6 categories such as HUMAN and LOCATION, a first step in a question answering system. For example, “What is the width of a football field?” is to be classified as a NUMBER. The corpus for this task is the UIUC dataset (Li and Roth, 2002) that consists of $5,542$ questions for training and 500 questions for test[3].

Tree kernels were shown to offer state of the art classification accuracy for this dataset. More details about tree kernels for question classification can be found in (Annesi et al., 2014). In this work, we are concerned with exploiting lexical and syntactic information only and therefore choose SSTK and PTKs for comparison with ASM and exclude Semantic Partial Tree Kernel (SPTK) that incorporates semantic word similarity (via clustering or word2vec (Goldberg and Levy, 2014)). In Table 5, we report the accuracy of these kernels for the question classification task.

4 Discussion

Evaluation over the two relation extraction tasks reveals that the ASM kernel outperforms both the tree kernels and the classical ASM rule based system. For a more general sentence classification task, we note that the ASM kernel performs competitively to tree kernels but not better. A study of the final scores attained also reveals that the relation extraction tasks are more difficult than general sentence classification tasks. We infer that the flexibility of the ASM kernel such as ability to handle edge labels and directions, is more advantageous in a relation extraction task than a general sentence classification task. This may be due to the fact that relation extraction is primarily focussed on the interaction between two entities in a sentence, which is best described by the edge labels on the shortest dependency path. In contrast, sentence classification is more general and considers the overall properties of a sentence.

Feature selection A closer study of the relation extraction tasks revealed that a simple linear classifier with bag of words and few syntactic features (the lemmas and POS tags of the neighbors of entity nodes in the dependency graph) outperforms any of the kernel methods discussed in this paper. These results are presented in Table 6. This observation suggests that kernel methods are likely to benefit by a simplification or pruning of their feature sets. The clearly defined feature space underlying the ASM kernel makes it amenable to intelligent feature selection techniques such as principal

[3]http://cogcomp.cs.illinois.edu/Data/QA/QC/

component analysis (PCA) that we plan to explore in future.

Semantic matching ASM relies on comparing properties of paths or graph walks, that are indexed by label pairs. In the simplest case, node labels are taken to be word lemmas instead of word tokens, to improve generalization across minute variations in word usage (such as “cured” and “curing”). We hypothesize that the generalizability of ASM can be further improved by choosing node labels on word classes. Node labels may be set to cluster ids, post word clustering. A semantic matching of lemmas (such as “cured” and “improved”), based on word semantics using distributional word similarities may allow for improved generalization (Saleh et al., 2014).

Task	Kernel	P	R	F1
Seedev	ASM Kernel	25.0	27.9	26.4
Seedev	Linear (hand crafted features)	30.0	34.9	**32.3**
CID	ASM Kernel	46.4	77.7	58.1
CID	Linear(hand crafted features)	54.8	81.5	**65.6**

Table 6: Comparison of ASM kernel with a linear classifier with hand crafted features.

5 Related Work

The closest work to ours is the classical ASM distance (Liu et al., 2013) that has been successfully used in several shared tasks (Kim et al., 2011). Tree kernels in NLP have been studied extensively in (Collins and Duffy, 2001). Relation extraction has been of particular importance within biomedical domain and has been studied in different contexts such as drug-drug interaction (Bjorne et al., 2011) and protein-protein interaction (Lan et al., 2009). Kernels that use constituent parses or dependency structures are studied in (Chowdhury et al., 2011; Airola et al., 2008) for the protein-protein interaction extraction. All path graph (APG) kernel (Airola et al., 2008) over dependency graph is a related work, that has different semantics as compared to ASM. The APG kernel considers all paths between a vertex pair and not just single shortest path as done in the ASM kernel. The primary feature in the APG kernel is the strength of connection between a vertex pair, which is computed as the product of edge weights along the path. Note that edge labels and not edge weights are the natural properties of a dependency graph. APG proposes modifications to the dependency graph to accommodate edge labels and heuristically driven assignment of edge weights to the dependency graph. An other recent approach in kernel design (Saleh et al., 2014), has been the efforts to include word similarity such as distributional word similarity given by word2vec (Goldberg and Levy, 2014). Incorporating semantic word similarity in ASM is likely to further improve its performance.

6 Summary and Conclusion

In this work, we defined a graph kernel from a previously studied Approximate Subgraph Matching (ASM) distance measure. We demonstrate the effectiveness of this new kernel by experimenting over standard datasets for question classification and relation extraction. Results indicate that the ASM kernel is of comparable performance to the tree kernels for sentence classification, but outperforms tree kernels in relation extraction tasks. We show the validity of the ASM kernel by deriving its feature space and illuminating the semantics of the kernel. Following on these steps, we identify several improvements to the ASM kernel that are likely to further boost its performance.

References

Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Salakoski. 2008. A graph kernel for protein-protein interaction extraction. In *Proceedings of the workshop on current trends in biomedical natural language processing*, pages 1–9. Association for Computational Linguistics.

Paolo Annesi, Danilo Croce, and Roberto Basili. 2014. Semantic compositionality in tree kernels. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM ’14, pages 1029–1038, New York, NY, USA. ACM.

Jari Bjorne, Antti Airola, Tapio Pahikkala, and Tapio Salakoski. 2011. Drug-drug interaction extraction from biomedical texts with SVM and RLS classifiers. *CEUR Workshop Proceedings*, 761:35–42.

Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods* *in Natural Language Processing*, pages 724–731. Association for Computational Linguistics.

Christopher JC Burges. 1998. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167.

Estelle Chaix, Bertrand Dubreucq, Abdelhak Fatihi, Dialekti Valsamou, Robert Bossy, Mouhamadou Ba, Louise Delger, Pierre Zweigenbaum, Philippe Bessires, Loc Lepiniec, and Claire Ndellec. 2016. Overview of the regulatory network of plant seed development (seedev) task at the bionlp shared task 2016. In *Proceedings of the 4th BioNLP Shared Task workshop*, Berlin, Germany, August. Association for Computational Linguistics.

Faisal Mahbub Chowdhury, Alberto Lavelli, and Alessandro Moschitti. 2011. A study on dependency tree kernels for automatic extraction of protein-protein interaction. In *Proceedings of BioNLP 2011 Workshop*, BioNLP '11, pages 124–133, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in neural information processing systems*, pages 625–632.

Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1034–1046. Association for Computational Linguistics.

Simone Filice, Giuseppe Castellucci, Danilo Croce, and Roberto Basili. 2015. Kelp: a kernel-based learning platform for natural language processing. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 19–24, Beijing, China, July. Association for Computational Linguistics and The Asian Federation of Natural Language Processing.

Yoav Goldberg and Omer Levy. 2014. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

Jin-Dong Kim, Sampo Pyysalo, Tomoko Ohta, Robert Bossy, Ngan Nguyen, and Junichi Tsujii. 2011. Overview of bionlp shared task 2011. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, pages 1–6. Association for Computational Linguistics.

Man Lan, Chew Lim Tan, and Jian Su. 2009. Feature generation and representations for protein-protein interaction classification. *Journal of Biomedical Informatics*, 42:866–872.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Haibin Liu, Lawrence Hunter, Vlado Kešelj, and Karin Verspoor. 2013. Approximate subgraph matching-based literature mining for biomedical events and relations. *PloS one*, 8(4):e60954.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *European Conference on Machine Learning*, pages 318–329. Springer.

I Saleh, Alessandro Moschitti, Preslav Nakov, L Màrquez, and S Joty. 2014. Semantic Kernels for Semantic Parsing. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 436–442.

Raimund Seidel. 1995. On the all-pairs-shortest-path problem in unweighted undirected graphs. *Journal of computer and system sciences*, 51(3):400–403.

Philippe Thomas, Mariana Neves, Illés Solt, Domonkos Tikk, and Ulf Leser. 2011. Relation extraction for drug-drug interactions using ensemble learning. *Training*, 4(2,402):21–425.

Chih-Hsuan Wei, Yifan Peng, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Jiao Li, Thomas C Wiegers, and Zhiyong Lu. 2015. Overview of the biocreative v chemical disease relation (cdr) task. In *Proceedings of the fifth BioCreative challenge evaluation workshop, Sevilla, Spain*.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (to appear)*.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network. In *COLING*, pages 2335–2344.