Sieve-Based Spatial Relation Extraction with Expanding Parse Trees

Jennifer D'Souza and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{jld082000,vince}@hlt.utdallas.edu

Abstract

A key challenge introduced by the recent SpaceEval shared task on spatial relation extraction is the identification of MOVELINKs, a type of spatial relation in which up to eight spatial elements can participate. To handle the complexity of extracting MOVELINKs, we combine two ideas that have been successfully applied to information extraction tasks, namely *tree kernels* and *multi-pass sieves*, proposing the use of an expanding parse tree as a novel structured feature for training MOVELINK classifiers. Our approach yields state-of-the-art results on two key tasks in SpaceEval.

1 Introduction

Spatial relation extraction is the task of determining the relation among a set of spatial elements. Although it has thus far received much less attention than temporal relation extraction, there has been a surge of interest in it in recent years, as evidenced by the organization of the three shared tasks on spatial relation extraction, namely the spatial role labeling tasks in 2012 (Kordjamshidi et al., 2012) and 2013 (Kolomiyets et al., 2013), as well as this year's SpaceEval task (Pustejovsky et al., 2015). The task has also evolved over the years, with new types of spatial elements and/or spatial relations being defined in each shared task. For instance, while the first two shared tasks have focused on extracting spatial relations between *stationary* objects, SpaceEval examines for the first time spatial relations on objects *in motion*.

Extracting spatial relations on objects in motion, or MOVELINKs, is very challenging. The challenge stems in part from the fact that a MOVELINK involves two *mandatory* participants (with roles $mover$ and $trigger$) and up to six *optional* participants (with other semantic roles). As an example, consider the MOVELINK that can be extracted from the sentence "John walked from Boston to Cambridge". As shown in Figure 1, this MOVELINK involves six spatial elements: "John" as the $mover$, "walked" as the $trigger$, "Boston" as the $source$, "Cambridge" as the $goal$, and "from" and "to" as the $motion_signal$s.

Figure 1: A MOVELINK example.

Given the complexity of MOVELINKs, any approach that attempts to *jointly* identify all the spatial elements involved in a MOVELINK and their roles is computationally infeasible. On the other extreme, one can tackle the task by identifying each element involved in a MOVELINK *independently* of the other elements. In fact, this is roughly the approach adopted by our participating system in SpaceEval (D'Souza and Ng, 2015), which achieved the best results in one of the SpaceEval tasks involving MOVELINK extraction. Specifically, this system trains one classifier for each *optional* role r to identify the filler for r in a MOVELINK independently of the other optional roles. Although this approach has achieved state-of-the-art performance, it is arguably not ideal: intuitively, dependencies exist among elements of different roles, and not capturing them may harm system performance.

Our goal in this paper is to advance the state of the art in spatial relation extraction, focusing on the extraction of MOVELINKs by addressing the aforementioned weakness. The key question is: how can we capture the dependencies among the spatial elements involved without sacrificing computational tractability? To address this question, we combine two ideas that have been successfully applied to a variety of information extraction tasks, namely multi-pass sieves (Raghu-

nathan et al., 2010; Lee et al., 2013) and tree kernels (Moschitti, 2004; Moschitti, 2006). Recall that a sieve-based approach is composed of a pipeline of *sieves* ordered by precision, where the decisions made by earlier sieves can be exploited by later sieves in order to incrementally construct a complex structure. When applied to MOVELINK identification, we can create a sieve for identifying each role, so that (1) spatial elements corresponding to different roles are incrementally added to a MOVELINK, and (2) earlier attachment decisions can be exploited as additional contextual information by later sieves. Hence, compared to a joint approach, a sieve-based approach achieves computational tractability by modeling *partial*, rather than *full* dependencies among the spatial elements.

While a sieve-based approach allows us to exploit additional contextual information provided by earlier sieves, we still have to specify how we encode such contextual information. Motivated by the successful application of tree kernels to relation extraction (e.g., Zelenko et al. (2003), Culotta and Sorensen (2004), Bunescu and Mooney (2005), Zhou et al. (2007)), we propose to (1) encode the *syntactic* context in which the spatial elements extracted by the sieves appear using a syntactic parse tree, and then (2) employ the tree as an (additional) structured feature for training the classifier associated with each sieve. This novel combination of sieves and tree-based structured features results in what we call an *expanding* parse tree. Specifically, as a spatial element for a MOVELINK is extracted by a (role-specific) sieve, it will be added to the structured feature for the classifier associated with the following sieve. In other words, the parse tree corresponding to the structured feature will keep expanding as we move along the sieves in the pipeline. This contrasts with previous applications of tree kernels, where a structured feature is created from a static parse subtree for extracting *exactly two* arguments involved in a relation. To our knowledge, this is the first attempt to combine sieves and parse trees to create expanding trees to extract complex relations involving multiple arguments.

2 Corpus and Task Definition

In this section, we introduce our corpus and the spatial relation extraction task. Owing to space limitations, we will only discuss those aspects that are relevant to the SpaceEval tasks we focus on.

2.1 The SpaceEval Corpus

We use as our corpus the SpaceEval training corpus, which is a subset of ISO-SpaceBank (Pustejovsky and Yocum, 2013). The corpus consists of 59 travel narratives annotated with seven types of spatial elements (Table 1) and three types of spatial relations (Table 2), following the ISO-Space (2012) annotation specifications. Different types of spatial elements have different attributes. The only attribute that is relevant to our work is *semantic type*, which is one of the attributes of a *spatial entity*. *Semantic type* expresses the type of the relation it triggers and can take one of three values: topological, directional, or both.

What is missing in Table 2 about spatial relations is that each element participating in a relation has a *role*. Each QSLINK/OLINK involves exactly three elements participating as $trajector$ (the object of interest), $landmark$ (the grounding location), and $trigger$ (the relation indicator). On the other hand, a MOVELINK has two fixed participants and up to six optional participants. The two mandatory MOVELINK participants are $mover$ (object in motion) and $trigger$ (verb denoting motion). Five of the optional participants express different aspects of the $mover$ in space, namely, $source$ (the spatial element at the beginning of the motion path), $midpoint$ (the spatial elements along the motion path), $goal$ (the spatial element at the end of the motion path), $path$ (the spatial element that reflects the path of motion), and $landmark$ (the grounding location). The sixth optional participant, $motion\_signal$, connects the spatial aspect to the $mover$. Note that all spatial relations are *intra-sentential*.

2.2 The Spatial Relation Extraction Task

Given a set of n spatial elements, the spatial relation extraction task aims to (1) determine whether the elements form a spatial relation of a particular type, and if so, (2) classify the roles of each participating element. For example, from the sentence "The cup is on the table", two relations can be extracted: QSLINK(cup$_{trajector}$, table$_{landmark}$, on$_{trigger}$) and OLINK(cup$_{trajector}$, table$_{landmark}$, on$_{trigger}$). As another example, from the sentence "John walked from Boston to Cambridge", a MOVELINK with participants John$_{mover}$, walked$_{trigger}$, from$_{motion_signal}$, Boston$_{source}$, to$_{motion_signal}$, and Cambridge$_{goal}$ can be extracted.

Type	Description
place	A geographic entity or region (e.g., *lakes*, *mountains*) or an administrative entity (e.g., *towns*, *countries*)
path	A location where the focus is on the potential for traversal (e.g., *road*)
spatial entity	A spatially relevant entity that is neither a *place* nor a *path* (e.g., *car*)
non-motion event	An event that does not involve movement but is directly related to another spatial element
motion event	A species of event that involves movement (e.g., *arrived*)
motion signal	A particle, preposition, verb, or adverb that encodes path or manner information about a *motion event*
spatial signal	A preposition/prepositional phrase that reveals the relationship between two locations (e.g., *north of*)

Table 1: Seven types of spatial elements in SpaceEval.

Relation	Description
QSLINK	Exists between stationary spatial elements with a regional connection. E.g., in "The cup is on the table", the regions of "cup" and "table" are *externally connected* and hence they are involved in a QSLINK.
OLINK	Exists between stationary spatial elements expressing their relative or absolute orientations. E.g., in the above sentence, "cup" and "table" are involved in an OLINK, which conveys that "cup" is oriented *above* "table".
MOVELINK	Exists between spatial elements in motion. E.g., in "John walked from Boston to Cambridge", there is a MOVELINK involving mover "John", motion verb "walked", source "Boston", and goal "Cambridge".

Table 2: Three types of spatial relations in SpaceEval.

3 Related Work

Broadly speaking, existing spatial relation extraction systems have adopted either a *pipeline* approach or a *joint* approach to these subtasks. Given a set of spatial elements, a pipeline spatial relation extraction system (1) extracts the $trigger$s, (2) determines whether a spatial relation exists between each extracted $trigger$ and each of the remaining spatial elements, and (3) classifies the role of each non-$trigger$ in each pair of spatially-related elements (Kordjamshidi et al., 2011; Bastianelli et al., 2013; Kordjamshidi and Moens, 2015).

The major weakness of pipeline approaches is that errors in trigger identification can propagate to the relation classification component, whose errors can in turn propagate to the role labeling component. To address this weakness, Roberts et al. (2012; 2013) investigated *joint* approaches. Given a set of spatial elements with an assignment of roles to each element, a joint spatial relation extraction system uses a binary classifier to determine whether these elements form a spatial relation with the roles correctly assigned to all participating elements. In other words, the classifier will output a 1 if and only if (1) the elements in the set form a relation and (2) their roles in the relation are correct. The systems participating in SpaceEval all seem to be in favor of joint approaches (D'Souza and Ng, 2015; Nichols and Botros, 2015; Salaberri et al., 2015).

4 Baseline System

To ensure that we have a state-of-the-art baseline system for spatial relation extraction, we employ our SpaceEval participating system (D'Souza and Ng, 2015), which achieved the best results on task 3a in the official SpaceEval evaluation.[1]

This Baseline system performs *joint* role labeling and relation classification using an ensemble of classifiers. Specifically, it trains one classifier for extracting QSLINKs and OLINKs, and seven classifiers for extracting MOVELINKs. Creating these eight classifiers permits (1) separating the treatment of MOVELINKs from QSLINKs and OLINKs (because the former involves objects in motion while the latter involve stationary objects); and (2) simplifying the extraction of MOVELINKs (because its optional participants are extracted *independently* of each other by these classifiers).

4.1 Training the Baseline Classifiers

In this subsection, we describe how we train the Baseline classifiers, which include one classifier for identifying QSLINKs and OLINKs (Section 4.1.1) and seven classifiers for identifying MOVELINKs (Section 4.1.2).

4.1.1 The LINK Classifier

We collapse QSLINKs and OLINKs to a single relation type, LINK, identifying these two types of links using the LINK classifier. To understand why we can do this, recall from Section 2.1 that in QSLINKs and OLINKs, the $trigger$ has to be a *spatial signal* element having a semantic type attribute. If its semantic type is topological, it triggers a QSLINK; if it is directional, it triggers an OLINK; and

[1] Since the official annotated test data is not available to us, we cannot compare our results with the shared task systems' official results, but comparing against a state-of-the-art baseline will enable us to determine whether our approach is better than the best existing spatial relation extraction system.

if it is both, it triggers both relation types. Hence, if a LINK is identified by our classifier, we can simply use the semantic type of the relation's $trigger$ to determine whether the relation is a QSLINK, an OLINK, or both.

We create training instances for training a LINK classifier as follows. Following the joint approach described above, we create one training instance for each possible role labeling of each triplet of distinct spatial elements in each sentence in a training document. The role labels assigned to the spatial elements in each triplet are subject to the following constraints: (1) each triplet contains a $trajector$, a $landmark$, and a $trigger$; (2) neither the $trajector$ nor the $landmark$ are of type *spatial signal* or *motion signal*; and (3) the $trigger$ is a *spatial signal*. These role constraints are derived from the data annotation scheme. Note that a LINK may have at most one *implicit* participant. For instance, the relation LINK(balloon$_{trajector}$, up$_{trigger}$) extracted from the sentence "The balloon went up" has an implicit $landmark$. To allow for implicit participants, from each training instance we have created thus far, we create three additional training instances, where exactly one of the three participants has the value IMPLICIT.

A training instance is labeled as positive if and only if the elements in the triplet form a relation and their roles are correct. As an example, for the QSLINK and OLINK sentence in Table 2, exactly one positive instance, LINK(cup$_{trajector}$, table$_{landmark}$, on$_{trigger}$), will be created.

Each instance is represented using the 31 features, which can be broadly divided into seven types: lexical, grammatical, semantic, positional, distance, entity attributes, and entity roles.[2] We train the LINK classifier using the SVM learning algorithm as implemented in the SVM^{light} software package (Joachims, 1999). To optimize classifier performance, we tune two parameters, the regularization parameter C and the cost-factor parameter J, to maximize F-score on the development data.[3] Since joint tuning of these parameters is computationally expensive, we employ a hill-climbing algorithm to find a local maximum, altering one parameter at a time to optimize F-score by holding the other parameter fixed.

4.1.2 The Seven MOVELINK Classifiers

If we adopted the aforementioned joint method as is for extracting MOVELINKs, each instance would correspond to an octuple of the form ($trigger_i$, $mover_j$, $source_k$, $midpoint_m$, $goal_n$, $landmark_o$, $path_p$, $motion\_signal_r$), where each participant in the octuple is either a distinct spatial element with a role or the NULL element (if it is not present in the relation). However, generating role permutations for octuples from all spatial elements in a sentence is computationally infeasible. For this reason, we simplify MOVELINK extraction as follows. First, we decompose the MOVELINK octuple into seven smaller tuples including one pair and six triplets. These seven tuples are: [1] ($trigger_i$, $mover_j$); [2] ($trigger_i$, $mover_j$, $source_k$); [3] ($trigger_i$, $mover_j$, $midpoint_m$); [4] ($trigger_i$, $mover_j$, $goal_n$); [5] ($trigger_i$, $mover_j$, $landmark_o$); [6] ($trigger_i$, $mover_j$, $path_p$); and [7] ($trigger_i$, $mover_j$, $motion\_signal_r$). Then, we create seven separate classifiers for identifying these seven MOVELINK tuples, respectively.

Using this decomposition for MOVELINK instances, we can generate instances for each classifier using the aforementioned joint approach as is. For instance, to train classifier [1], we generate pairs of the form ($trigger_i$, $mover_j$), where $trigger_i$ and $mover_j$ are spatial elements proposed as a candidate $trigger$ and a candidate $mover$, respectively. Positive training instances are those ($trigger_i$, $mover_j$) pairs annotated as being part of a MOVELINK in the training data, while the rest of the candidate pairs are negative training instances. The instances for training the remaining six classifiers are generated similarly.

As in the LINK classifier, we enforce global role constraints when creating training instances for the MOVELINK classifiers. Specifically, the roles assigned to the spatial elements in each training instance of each MOVELINK classifier are subject to six constraints: (1) the $trigger$ has type *motion event*; (2) the $mover$ has type *place*, *path*, *spatial entity*, or *non-motion event*; (3) the $source$, the $goal$, and the $landmark$ can be NULL or have type *place*, *path*, *spatial entity*, or *non-motion event*; (4) the $midpoint$ can be NULL or have type *place*, *path*, or *spatial entity*; (5) the $path$ can be NULL or have type *path*; and (6) the $motion\_signal$ can

[2] Space limitations preclude a description of these features. See D'Souza and Ng (2015) for details.

[3] C is chosen from the set {0.01, 0.05, 0.1, 0.5, 1.0, 10.0, 50.0, 100.0}, and J is chosen from the set {0.01, 0.05, 0.1, 0.5, 1.0, 2.0, 4.0, 6.0}. All other learning parameters are set to their default values. In particular, a linear kernel is used.

be NULL or have type *motion signal*.

Our method for decomposing the octuple by role can be justified as follows. Since $trigger$ and $mover$ are mandatory MOVELINK participants, we have a classifier for classifying this core aspect of a MOVELINK. The next six classifiers, [2] to [7], aim to improve the core MOVELINK extraction by exploiting the contextual dependencies with each of its unique spatial aspects, namely $source$, $midpoint$, $goal$, $landmark$, $path$, and $motion\_signal$.

As an example, for the MOVELINK sentence in Table 2, we will create three positive instances: (John$_{mover}$, walked$_{trigger}$) for classifier [1], (John$_{mover}$, walked$_{trigger}$, Boston$_{source}$) for classifier [2], and (John$_{mover}$, walked$_{trigger}$, Cambridge$_{goal}$) for classifier [4].

We represent each training instance using the 31 features that were used to train the LINK classifier. We train each of the MOVELINK classifiers using SVM^{light}. We tune the C and J parameters to maximize F-score on the development data using the hill-climbing algorithm described earlier.[4]

4.2 Applying the Baseline Classifiers

After training, we apply the resulting classifiers to classify the test instances, which are created in the same way as the training instances. As noted before, each LINK extracted from a test document by the LINK classifier is further qualified as QSLINK, OLINK, or both based on the semantic type of its $trigger$. The MOVELINKs are extracted from a test document by combining the outputs from the seven MOVELINK classifiers.

There is a caveat, however: different MOVELINK classifiers can make conflicting decisions. For instance, classifier [1] might misclassify (John$_{mover}$, walked$_{trigger}$) as negative, whereas classifier [2] might correctly classify (John$_{mover}$, walked$_{trigger}$, Boston$_{source}$) as positive. To resolve these conflicting decisions, we give preference to positive decisions, meaning that in this case we will posit "John" and "walked" as having the roles of $mover$ and $trigger$ respectively. There are two more sources of conflicts. First, a spatial element may be assigned different roles for a given MOVELINK by different classifiers. Second, the global constraint that each MOVELINK can have at most one $source$, at most one $goal$, and at most one $landmark$ can be violated. To resolve these conflicts, we select for each spatial element the role that was predicted with highest confidence by the SVM classifiers subject to the global constraint.[5]

5 Our Multi-Pass Sieve Approach

In this section, we describe two methods for employing sieves for extracting spatial relations.

5.1 Using Sieves without Trees

To motivate our first method, recall that the Baseline resolves conflicting decisions in a heuristic manner. For instance, it prefers positive to negative decisions, effectively favoring recall over precision for spatial relation extraction. It is not clear whether this ad-hoc decision is good or not. As another example, when more than one role is assigned to the same spatial element in a MOVELINK, it favors the role associated with the highest SVM confidence. This, however, is also an ad-hoc decision: recall that each classifier's parameters are tuned independently of the others, so different confidence values assigned by different classifiers are not directly comparable.

Employing sieves for spatial relation extraction obviates the need for such ad-hoc decisions. In our implementation, we have eight sieves, each of which corresponds to exactly one of the eight classifiers employed by the Baseline. Recall from the introduction that these sieves are ordered as a pipeline. So, whenever a conflict arises, earlier sieves' decisions have precedence over later sieves' decisions. Returning to the conflicting decisions mentioned before, if sieve 1 misclassifies (John$_{mover}$, walked$_{trigger}$) as negative, whereas sieve 2 correctly classifies (John$_{mover}$, walked$_{trigger}$, Boston$_{source}$) as positive, then we will posit that no MOVELINK exists between "John" and "walked" because we have more confidence in sieve 1's decision than sieve 2's decision. As another example, if two classifiers assign different roles to the same spatial element, then we will choose the role assigned by the classifier associated with the earlier sieve.

Given the above discussion, it should be clear that the ordering of the sieves is important. Typically, sieves are ordered by *precision*, with the hope of reducing the number of erroneous deci-

[4] See Footnote 3 for the set of values of C and J used for parameter tuning.

[5] We use the distance from the hyperplane as a measure of an SVM classifier's confidence.

sions passed from the earlier sieves to the later sieves (e.g., Raghunathan et al. (2010), Lee et al. (2013), Chambers et al. (2014)). Motivated by this observation, we order the sieves as follows. We set sieve 0 to be the LINK classifier and sieve 1 to be the ($trigger$, $mover$) classifier, and then order the remaining sieves by precision. Specifically, we compute the precision of each sieve on the development data, then add sieves into the pipeline in decreasing order of precision.

5.2 Using Sieves with Trees

Next, we describe our second method for applying sieves to spatial relation extraction. To motivate this method, recall that one important property of a sieve-based approach is that later sieves can exploit earlier sieves' decisions when making their own decisions. However, our first method of using sieves makes limited use of the decisions made by earlier sieves. In particular, while each sieve exploits the knowledge of whether a spatial element has been assigned a role by an earlier sieve, it does not exploit the knowledge of what the role is.

Our second method exploits the role decisions made by earlier sieves, but another question arises: how can we encode these role decisions so that they can be best exploited by later sieves? One possibility is to employ them as additional features for training the classifiers associated with later sieves. Motivated by previous work on tree kernels for relation extraction, we employ parse trees as a *structured* feature to encode the syntactic relationships among the roles extracted so far for a given MOVELINK.

We create the structured feature as follows. To strike a better balance between having a rich representation of the context surrounding a spatial relation and improving the learner's ability to generalize, we extract a *subtree* from a parse tree and use it as the value of the structured feature. Specifically, given relation candidate triplet (e_1,e_2,e_3), where spatial elements e_1, e_2, and e_3 are posited in roles r_1, r_2, and r_3, respectively, and the associated syntactic parse tree T, we extract our parse subtree from T as follows. First, we identify the smallest subtree that covers all three spatial elements and call its root r. Second, for each path from each spatial element to r, we include in the parse subtree all the nodes that lie on the path and their immediate children. Third, we simplify the subtree by removing the POS nodes above each spatial element, effectively attaching it to its grandparent. Finally, for better generalization, we replace each spatial element with its role.

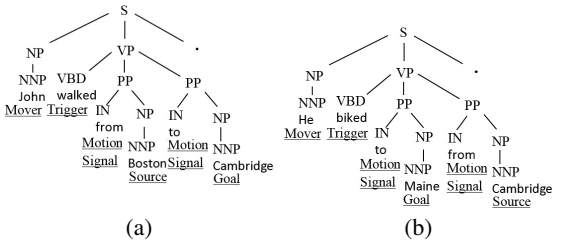


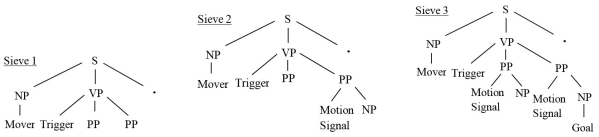
Figure 2: Syntactic parse trees for two example sentences containing MOVELINKs.

Since this subtree centers on a spatial relation, we call it a *spatial relation centered tree* (SRCT). As mentioned before, we will use SRCTs in combination with our sieve-based approach. This results in a novel application of structured features: to our knowledge, all trees that were previously used as structured features were *static*. By contrast, SRCTs expand as we move along the sieve pipeline. We will discuss examples of how to create SRCTs below.

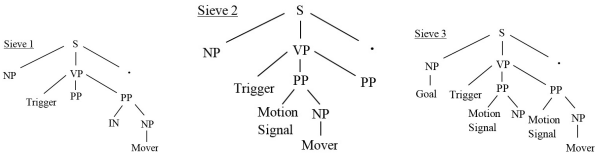
Creating training instances with structured features is straightforward. Recall that a structured feature is used as an additional feature when training each classifier. In other words, it will be used in combination with the original 31 features. The training instance creation method used by the Baseline remains unchanged. All we need to do is to add a SRCT as a structured feature to each training instance.

Consider the sentence "John walked from Boston to Cambridge", whose parse tree is shown in Figure 2(a). Since only one MOVELINK can be extracted from it, only one positive training instance will be created for each classifier. Assume that sieve 1 contains the ($trigger$, $mover$) classifier; sieve 2 contains the ($trigger$, $mover$, $motion\_signal$) classifier; and sieve 3 contains the ($trigger$, $mover$, $goal$) classifier. Figure 3(a) shows the SRCTs used in the corresponding positive instances, while Figure 3(b) shows the SRCTs associated with randomly chosen negative instances used to train the classifiers in these sieves.

To train a classifier on instances containing the original 31 features *and* SRCTs, we employ SVM$^{light-TK}$ (Moschitti, 2004; Moschitti, 2006), which (1) trains an SVM classifier using the 31 features with a linear kernel; (2) trains an SVM classifier using only the SRCTs with a convolution

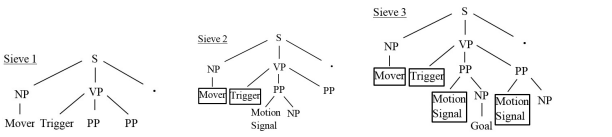


(a) SRCTs associated with positive training instances in sieves 1, 2, and 3.

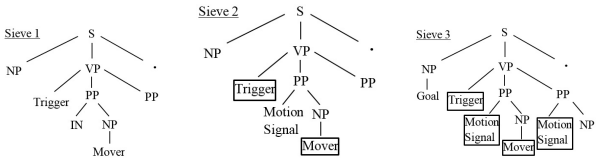


(b) SRCTs associated with randomly chosen negative training instances in sieves 1, 2, and 3.

Figure 3: SRCTs associated with training instances created for the sentence in Figure 2a.



(a) SRCTs associated with correctly classified test instances in sieves 1, 2, and 3.



(b) SRCTs associated with incorrectly classified test instances in sieves 1, 2, and 3.

Figure 4: SRCTs associated with test instances for the sentence in Figure 2b.

kernel; and (3) combines these two kernels using a composite kernel. Specifically, we define composite kernel K_c for combining linear kernel K_l and convolution kernel K_t as follows:

$$K_c(F_1, F_2) = K_l(F_1, F_2) + T \cdot K_t(T_1, T_2),$$

where F_1 and F_2 are the set of 31 features from two training instances, T_1 and T_2 are their SRCTs, and T is the combination parameter. We employ the hill-climbing algorithm described before to tune the C, J and T parameters to maximize F-score on the development data.[6]

The test instances with structured features are created in the same way as the training instances, with one notable difference. Note that roles are used to create structured features. While they are available in training, they are not available in the test documents. Hence, to create SRCTs for a test instance, we have to employ the roles *predicted* by preceding sieves. This is precisely how we exploit the role decisions made by earlier sieves in later sieves. This also explains why the SRCTs expand: more and more roles will be attached to a SRCT as it passes through the sieve pipeline.

As an example, consider the test sentence "He biked to Maine from Cambridge", whose parse tree is shown in Figure 2(b). Figure 4(a) shows the SRCTs generated when each sieve makes the correct decisions. Specifically, sieve 1 correctly identifies "He" as the $mover$ and "biked" as the $trigger$. The roles (correctly) extracted by sieve 1 (shown in boxes) are incorporated into the SRCT created in sieve 2. Similarly, the $motion_signal$s (correctly) extracted by sieve 2 are incorporated into the SRCT created in sieve 3. Figure 4(b) shows the SRCTs generated for misclassified instances. Assume that sieve 1 misclassifies the test instance underlying the SRCT (as positive). As we can see, this mistake is propagated to the SRCTs generated in later sieves.

Note that the precision of a sieve may change with the addition of SRCTs. For this reason, the sieves need to be reordered using the algorithm described in the previous subsection.

6 Evaluation

6.1 Experimental Setup

SpaceEval tasks. We evaluate our approach in tasks 1d and 3a of SpaceEval. These two tasks evaluate a system's ability to extract ($trajector$, $landmark$, $trigger$) triplets in QSLINKs and OLINKs as well as ($trigger$, $mover$) pairs in MOVELINKs using gold spatial elements (1d) and automatically extracted spatial elements (3a). To extract the spatial elements needed for task 3a, we follow Bastianelli et al.'s (2013) sequence labeling approach, except that we train the sequence labeler using a CRF rather than an HMM.[7]

Dataset. Since the annotated test set used in SpaceEval's official evaluation is not available

[6] To tune T, we attempted values between 0 and 2 in increments of 0.2. To tune C and J, we attempted the values specified in Footnote 3.

[7] We train two CRF models using CRF++ (`https://taku910.github.io/crfpp/`), one to extract *motion signal*s and the other to extract the remaining six types of spatial elements (see Table 1). The reason is that *motion signal* is the only type of spatial element that can overlap with other types. When predicting *spatial signal*s, we also predict their semantic types, since the LINK classifier needs this attribute to distinguish between QSLINKs and OLINKs (see Section 4.1.1). To increase recall, we use the 10-best outputs returned by the CRFs as candidate spatial elements.

	QSLINK						OLINK						MOVELINK						**OVERALL**		
	False			True			False			True			False			True					
	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F
Baseline	99.5	99.4	99.5	46.9	48.9	47.9	100	99.4	**99.7**	50.3	100	**66.9**	91.3	99.8	95.3	84.8	61.5	71.3	78.8	84.8	81.7
Sieve	99.5	99.4	99.5	46.9	48.9	47.9	100	99.4	**99.7**	50.3	100	**66.9**	94.7	99.8	97.2	79.4	72.1	75.5	78.5	86.6	82.3
+ SRCTs	99.8	99.4	**99.6**	43.1	66.4	**52.3**	100	99.4	**99.7**	43.7	100	60.8	97.1	99.8	**98.4**	77.3	82.3	**79.7**	76.8	91.2	**83.4**

(a) Results obtained using gold spatial elements.

	QSLINK						OLINK						MOVELINK						**OVERALL**		
	False			True			False			True			False			True					
	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F
Baseline	99.8	99.9	**99.9**	28.5	11.5	16.4	100	99.9	**100**	31.2	100	**47.5**	92.3	50.0	64.9	54.0	17.1	26.0	67.6	63.1	65.3
Sieve	99.8	99.9	**99.9**	28.5	11.5	16.4	100	99.9	**100**	31.2	100	**47.5**	96.1	50.0	**65.8**	42.5	25.3	31.7	66.4	64.4	65.4
+ SRCTs	99.8	99.9	**99.9**	28.3	12.9	**17.8**	100	99.9	**100**	31.2	100	**47.5**	94.7	50.0	65.5	56.2	24.5	**34.2**	68.4	64.6	**66.4**

(b) Results obtained using extracted spatial elements.

Table 3: Results for extracting spatial relations.

to us at the time of writing, we conduct our evaluation on the SpaceEval training corpus, which contains 1890 spatial relations (886 QSLINKs, 225 OLINKs, and 779 MOVELINKs) and 1139 MOVELINK optional roles (95 $source$s, 65 $midpoint$s, 310 $goal$s, 77 $landmark$s, 100 $path$s, and 492 $motion_signal$s). We partition the 59 narratives in the corpus into five folds and report five-fold cross-validation results. In each fold experiment, we employ three folds for training, one fold for development, and one fold for testing.

Evaluation metrics. To evaluate the results for the two SpaceEval tasks, we employ the official SpaceEval scoring program, which reports results in terms of recall, precision, and F-score on the three types of spatial relations in isolation and in combination. To evaluate the results for extracting MOVELINK optional roles, we compute the recall, precision, and F-score for each role.

6.2 Results and Discussion

Tables 3a and 3b show the spatial relation extraction results for three systems — the Baseline (row 1), our Sieve approach without SRCTs (row 2), and our Sieve approach with SRCTs (row 3) — obtained using gold and extracted spatial elements, respectively.

The official scoring program reports spatial relation extraction results in terms of recall (R), precision (P), and F-score (F). For QSLINKs and OLINKs, it reports results on (1) extracting spatially-related ($trajector$, $landmark$, $trigger$) triplets (see the "True" columns) and (2) identifying that no spatial relation exists among a (candidate $trajector$, candidate $landmark$, candidate $trigger$) triplet (see the "False" columns). For MOVELINKs, it reports results on (1) extracting spatially-related ($trigger$, $mover$) pairs (see the "True" columns) and (2) identifying that no spatial relation exists between a (candidate $trigger$, candidate $mover$) pair (see the "False" columns). Since the pairs/triplets without links considerably outnumber those with links, the OVERALL scores are dominated by the performances on "False", which, as expected, are high.

Of particular interest are the MOVELINK scores under the "True" columns. When gold spatial elements are used, Sieve significantly outperforms Baseline ($p < 0.001$) owing to substantial gains in precision with smaller losses in recall.[8] Adding SRCTs to Sieve further boosts performance significantly ($p < 0.005$). As we can see, Sieve+SRCTs outperforms Baseline by 8.4% absolute F-score on extracting ($trigger$, $mover$) pairs. With respect to the OVERALL score, which also takes into account QSLINKs and OLINKs, Sieve+SRCTs outperforms Baseline significantly by 1.7% absolute F-score. Similar trends can be observed for the results obtained using extracted spatial elements. Note that Baseline and Sieve have the same QSLINK and OLINK results because the LINK classifier is associated with the first sieve.

Tables 4a and 4b show the results on extracting MOVELINK optional roles using gold and extracted spatial elements, respectively. When gold elements are used, Sieve insignificantly outperforms Baseline on $source$, $midpoint$, and $motion\_signal$. When used with SRCTs, Sieve insignificantly outperforms Baseline on $source$, $midpoint$, $path$, and $motion\_signal$. Overall, Sieve+SRCTs insignificantly outperforms Baseline by 3.0% absolute F-score.[9] While the results

[8] All statistical significance tests are paired t-tests, with p set to 0.05 unless otherwise stated.

[9] A closer examination of the results reveals why the improvement is insignificant: since many roles occur infrequently in the corpus, the parameters learned from the devel-

	source			*midpoint*			*goal*			*landmark*			*path*			*motion-signal*			**OVERALL**		
	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F
Baseline	34.7	37.1	35.9	43.1	26.9	33.1	46.1	73.7	**56.8**	19.5	31.9	**24.2**	44.0	62.0	51.5	72.6	67.4	69.9	54.4	59.9	57.0
Sieve	29.5	62.2	**40.0**	30.8	39.2	34.5	43.6	79.4	56.3	13.0	37.0	19.2	36.0	69.2	47.4	67.5	74.8	71.0	49.3	71.1	58.2
+ SRCTs	23.2	100	37.6	30.8	71.4	**43.0**	41.3	85.3	55.7	5.2	66.7	9.6	40.0	85.1	**54.4**	64.0	84.5	**72.8**	46.5	84.5	**60.0**

(a) Results obtained using gold spatial elements.

	source			*midpoint*			*goal*			*landmark*			*path*			motion signal			**OVERALL**		
	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F
Baseline	25.3	12.0	16.3	35.4	8.9	14.3	38.7	15.2	21.8	18.2	2.5	4.4	14.0	12.9	13.4	27.7	24.0	25.7	29.1	13.3	18.2
Sieve	10.5	23.3	14.5	20.0	20.2	20.2	30.3	24.1	26.9	5.2	3.3	4.1	10.0	20.0	13.3	27.4	25.3	**26.3**	23.4	22.1	22.7
+ SRCTs	15.8	51.7	**24.2**	16.9	34.4	**22.7**	32.3	25.4	**28.4**	5.2	8.3	**6.4**	12.0	18.5	**14.6**	28.1	22.9	25.2	24.6	23.9	**24.3**

(b) Results obtained using extracted spatial elements.

Table 4: Results for extracting MOVELINK optional roles.

obtained using extracted elements exhibit different trends, Sieve+SRCTs's OVERALL improvement of 6.1% absolute F-score over Baseline is significant ($p < 0.001$).

Table 5 shows the results of these systems on extracting entire MOVELINKs, where a MOVELINK is considered correctly extracted if *all* of its participating elements and their roles are correct. Note that none of the SpaceEval tasks employ this stringent but informative evaluation measure. As we can see, Sieve+SRCTs insignificantly outperforms Baseline by 2.3–3.0% absolute F-score, regardless of whether gold or extracted elements are used.

6.3 Error Analysis

In this subsection, we analyze the errors made by the best-performing system, Sieve+SRCTs, with respect to the extraction of MOVELINKs given our focus on this type of spatial relation.

Extracting ($mover$, $trigger$) pairs. The major source of recall error stems from the system's inability to extract $mover$s that are unseen in the training data. This error could be addressed using a named entity recognizer and WordNet categories related to people, places, animals, etc. The major source of precision error arises from *missing* gold annotations. Consider the sentence "I found myself biking..." Our system correctly extracted "biking" as the $trigger$ and "I" as the $mover$, but was considered wrong because "myself", not "I", was annotated as the $mover$ in the gold standard.

Extracting optional roles. A major source of recall/precision error stems from the system's inability to exploit contextual cues that are reliable indicators of a particular role. For instance, a statistical analysis of the training data reveals that $source$s are commonly preceded by prepositions such as "from" and "or", whereas $goal$s are commonly associated with verbs such as "return", "visit", "arrive", and "reach". This problem could be addressed by encoding these cues explicitly as additional features for training the role-specific classifiers. Another source of recall error can be attributed to the lack of background knowledge. Consider the sentence "We had only 70 more km to Cluj taking this way, but if getting back to Ciucea and on to Cluj the normal way would have been 25 km longer". Despite correctly extracting "Cluj" as the $goal$, the system failed to extract "Ciucea" as the $midpoint$. This problem could be alleviated by exploiting geographical knowledge concerning these cities in external knowledge sources such as Wikipedia.

	Gold			Extracted		
	R	P	F	R	P	F
Baseline	40.6	50.4	45.0	21.7	15.3	18.0
Sieve	40.7	50.4	45.0	20.9	14.9	17.4
+ SRCTs	40.5	59.0	**48.0**	23.3	18.0	**20.3**

Table 5: Results for extracting entire MOVELINKs using gold and extracted elements.

7 Conclusions

We have examined the under-studied task of spatial relation extraction, focusing on spatial relations of objects in motion. Our approach exploited expanding parse trees, which resulted from a novel combination of multi-pass sieves and tree kernels, achieving state-of-the-art results on two key SpaceEval tasks. To facilitate comparison with future work on this task, we released the source code of our spatial relation extraction system.[10]

Acknowledgments

We thank the three anonymous reviewers for their comments. This work was supported in part by NSF Grants IIS-1147644 and IIS-1219142.

opment data are not necessarily the same as those that yield the best results on the test data.

[10] See our website at `http://www.hlt.utdallas.edu/~jld082000/spatial-relations/` for details.

References

Emanuele Bastianelli, Danilo Croce, Daniele Nardi, and Roberto Basili. 2013. Unitor-HMM-TK: Structured kernel-based learning for spatial role labeling. In *Proceedings of the 7th International Workshop on Semantic Evaluation*, pages 573–579.

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

Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. 2014. Dense event ordering with a multi-pass architecture. *Transactions of the Association for Computational Linguistics*, 2:273–284.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics, Main Volume*, pages 423–429.

Jennifer D'Souza and Vincent Ng. 2015. UTD: Ensemble-based spatial relation extraction. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 862–869.

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.

Oleksandr Kolomiyets, Parisa Kordjamshidi, Steven Bethard, and Marie-Francine Moens. 2013. SemEval-2013 Task 3: Spatial role labeling. In *Proceedings of the 7th International Workshop on Semantic Evaluation*, pages 255–266.

Parisa Kordjamshidi and Marie-Francine Moens. 2015. Global machine learning for spatial ontology population. *Web Semantics: Science, Services and Agents on the World Wide Web*, 30(C):3–21.

Parisa Kordjamshidi, Martijn Van Otterlo, and Marie-Francine Moens. 2011. Spatial role labeling: Towards extraction of spatial relations from natural language. *ACM Transactions on Speech and Language Processing*, 8(3):4.

Parisa Kordjamshidi, Steven Bethard, and Marie-Francine Moens. 2012. SemEval-2012 Task 3: Spatial role labeling. In *Proceedings of the 6th International Workshop on Semantic Evaluation*, pages 365–373.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916.

Alessandro Moschitti. 2004. A study on convolution kernels for shallow statistic parsing. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics, Main Volume*, pages 335–342.

Alessandro Moschitti. 2006. Making tree kernels practical for natural language processing. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 113–120.

Eric Nichols and Fadi Botros. 2015. SpRL-CWW: Spatial relation classification with independent multi-class models. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 895–901.

James Pustejovsky and Zachary Yocum. 2013. Capturing motion in ISO-SpaceBank. *Proceedings of the 9th Joint ISO - ACL SIGSEM Workshop on Interoperable Semantic Annotation*, pages 25–34.

James Pustejovsky, Jessica Moszkowicz, and Marc Verhagen. 2012. A linguistically grounded annotation language for spatial information. *TAL*, 53(2):87–113.

James Pustejovsky, Parisa Kordjamshidi, Marie-Francine Moens, Aaron Levine, Seth Dworman, and Zachary Yocum. 2015. SemEval-2015 Task 8: SpaceEval. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 884–894.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nate Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 492–501.

Kirk Roberts and Sanda M. Harabagiu. 2012. UTD-SpRL: A joint approach to spatial role labeling. In *Proceedings of the 6th International Workshop on Semantic Evaluation*, pages 419–424.

Kirk Roberts, Michael A. Skinner, and Sanda M. Harabagiu. 2013. Recognizing spatial containment relations between event mentions. In *Proceedings of the 10th International Conference on Computational Semantics – Long Papers*, pages 216–227.

Haritz Salaberri, Olatz Arregi, and Beñat Zapirain. 2015. IXAGroupEHUSpaceEval: (X-Space) A WordNet-based approach towards the automatic recognition of spatial information following the ISO-Space annotation scheme. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 856–861.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.

GuoDong Zhou, Min Zhang, DongHong Ji, and QiaoMing Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 728–736.