The MSR Systems for Entity Linking and Temporal Slot Filling at TAC 2013

Silviu Cucerzan
Microsoft Research
1 Microsoft Way
Redmond, WA 98052
silviu@microsoft.com

Avirup Sil *
Computer and Information Sciences
Temple University
Philadelphia, PA 19122
avi@temple.edu

Abstract

The paper describes the two systems from MSR that participated in the English Entity Linking and Temporal Slot Filling (TSF) tasks at TAC 2013. The entity linking system is built by using the same framework and architecture as the previous iterations that participated in the TAC 2011 and 2012 evaluations; therefore, its description focuses on the components that are novel with reference to those systems. The MSR system that addresses the newly introduced TSF task, employs a distant-supervision framework, in which language models for each targeted type of relation are built based on a corpus of sentences automatically extracted from Wikipedia text as guided by Wikipedia infobox data. For each of the TAC tracks addressed, we submitted two runs, which obtained the highest scores in their corresponding evaluations.

1 Introduction

Microsoft Research (MSR) has participated in the TAC evaluations for English entity linking since 2011 by employing the entity extraction and linking system developed by Silviu Cucerzan, first discussed in (Cucerzan, 2007). This year, in addition to participating in the evaluation for the same task by adapting the latest version of this system for the TAC format of the entity liking evaluation, MSR submitted for evaluation a system for the newly-introduced temporal slot filling task. This new system was developed as a research internship project by Avirup Sil and Silviu Cucerzan. While the two systems employ relatively independent approaches for addressing the two targeted tasks, they both start from the premise that richness of the community-built Wikipedia collection, whether semantic, lexical, syntactic, or structural, is a key enabler in achieving state-of-the-art performance for many NLP and IR task.

Section 2 of the paper describes the entity linking system that participated in this TAC evaluation. Two runs were submitted to TAC 2013 for the English entity linking task of the Knowledge Base Population track. They were generated by a system employing the same general architecture as in TAC 2011 and 2012, which employed two different machine learning methods employed for ranking the candidate entity disambiguations: a linear model trained with logistic regression, and a boosted-trees-based ranker. In addition to improved derivation of the knowledge base from Wikipedia and improved feature computation in comparison to the system from last year, the 2013 system employs additionally local features, entity triggers, and entity-type contexts, as derived from the Wikipedia collection.

Section 3 of the paper describes the temporal slot filling system built for participating in the TAC evaluation. Because of the lack of annotated data available in this first evaluation for the task, the system employs distant supervision for training by using Wikipedia infoboxes in conjunction with Wikipedia text as a semi-structured resource. From this automatically generated corpus, consisting of sentences from Wikipedia, we built language models for each type of relation, which were then used to extract temporal information from the targeted test documents.

* This research was carried out during an internship at Microsoft Research.

2 Entity Linking

2.1 Task Description

The entity linking task in the TAC evaluations, first introduced by McNamee and Dang (2009), consists of mapping given strings in text documents to entities from a knowledge base with over 818,000 entries, which was derived from the Wikipedia dump from October 2008. A large percentage of the target strings in the TAC queries account for entities that are not present in the reference collection, and should be mapped to NIL. For such cases, the task definition requires that all instances of strings that refer to the same unknown entity in the given query set be grouped together by providing an additional identifier (in the form of NILxxxx), which can be seen as a requirement to cluster all queries in the entity space regardless of whether the entities they mention belong to the reference knowledge base or not. Starting with 2012, the input queries also provide positional information for the targeted strings in the form of UTF-8 character offsets, which allows the evaluation to depart from the one-sense-per-discourse assumption (Gale et al., 1992b). The source document collection for the 2013 Entity Linking evaluation is composed of the English Gigaword Fifth Edition, the TAC 2012 KBP Source Corpus Additions Web Documents, and a new (for 2013) corpus of approximately half a million discussion forum posts.

2.2 System Architecture

The general architecture of the MSR entity linking system that participated in TAC 2011 and TAC 2012 has been preserved. In addition to improved derivation of the knowledge base and improved feature computation compared to the systems from previous years, the 2013 system employs for disambiguation additional local features, entity triggers, and entity contexts derived from Wikipedia. The two runs submitted for evaluation in TAC 2013 make use of two different machine learning methods for ranking the entity candidate for the given surface forms: a linear model trained with logistic regression, and a boosted-tree ranker.

The system takes as input a text document and attempts to identify and disambiguate/link to Wikipedia all entity mentions in the document. The output of this analysis process is a list of entities (identified by their canonical Wikipedia name) together with lists of the surface forms extracted from the document that are mapped by the system to each of those entities. To perform the TAC entity linking task, the system matches the target name string from the TAC query against the output surface forms extracted from the target document. In the matching process, the target name can be identical to one or more of the surface forms extracted from text, can be a substring of one or more of the extracted surface forms, or a superstring of one or more of those forms. The entities corresponding to all matched surface forms are ranked based on the type of match and frequency of the surface form, as well as the distance to the provided offset (because the system may not preserve the original input text because of the text normalization performed as a pre-processing step, it may not be possible to identify with certainty the targeted instance of the string as given by the input offset). The top-ranked entity is then returned as the answer. This strategy allows the system use its own boundary detection in conjunction with the disambiguation method to identify the most appropriate surface forms in the text globally, including the identification of entities mentioned by substrings and superstrings of the given target string. When no such match is found, the target document is processed a second time while enforcing that the exact target name string is a surface form to be disambiguated.

The system employs vectorial representations of entities in several different spaces (such as topics, contexts, geo-coordinates) on one hand, and probability distributions for mapping *known* surface forms (as extracted from the Wikipedia collection) to entities on the other. The analysis of an input text is done in three stages, with the following main roles:

- text normalization and sentence breaking;
- surface form boundary detection;
- document model construction and entity disambiguation.

The processing steps in these stages follow the approaches presented in the descriptions of the MSR systems that participated in the TAC 2011 and 2012 evaluations, with the text normalization stage basically unchanged, and the entity boundary detection following a similar process for accounting for multiple possible surface form boundaries at a certian position in text through composite surfaces until the disambiguation stage.

The document model construction and entity disambiguation stage also follow the same approaches as the previous iterations of the MSR system. For each possible disambiguation associated with a surface form, multiple features are computed. Some of these features, denoted as *observable* are computed directly, based on the similarity between the Wikipedia information associated with the candidate entities (such as context vectors) and the target document. Other features, called *latent*, are computed based on the aggregated *document model*, and measure the similarity of the information associated with the candidate entities for each surface form and the document representation obtained by aggregating the information from all possible disambiguations (candidate entities) for all surface forms in the given document. In the final stage, the entity assignment is calculated for each surface form as the argmax of either a logistic regression model (run 1) or the score of a boosted-tree ranker (run 2), which employ the features computed as discussed above for all candidate entity disambiguations. The training process for the two models is strictly based on the Wikipedia collection (as discussed in Section 2.3), with the TAC datasets available employed for validation. The most notable changes with respect to the previous systems refer to the addition of local features, lexical contexts, and entity triggers. These are discussed in more detail further.

2.2.1 Local Features

The 2013 MSR system departs from the one sense per discourse paradigm, and allows identical surface forms in text to be disambiguated differently based on local context.

For each surface form in the text, the system computes in addition to global features, which are derived from the whole document, a subset of local features, which are derived only from the paragraph to which the targeted surface form belongs. This subset comprises contexts, topics, and triggers. The local feature for context is computed in a similar manner to the corresponding global feature: the Wikipedia-based context vector for an entity candidate is directly matched against the text of the paragraph. However, in the case of the two other local features, for which their global counterparts are computed based on the document model, the computation takes into account local lexico-syntactic patterns, based on observations/justifications similar to those recently published by (Cheng and Roth, 2013). Pairs or sets of entities that are in conjunctive or possessive constructions, modify one another, or are part of enumerations, are given a preferential role in computing the local trigger and topic matching features.

2.2.2 Lexical Contexts

The 2013 system uses two new features for *local contexts*, defined as the immediate left and right context of a surface form, by employing sets of hypothesized entity type labels for entities as well as distributions over entity type labels for lexical contexts, similar to those employed in word sense disambiguation and context-sensitive spelling correction, e.g., (Gale et al., 1992a) and (Golding, 1995).

During the derivation of the knowledge base from the Wikipedia dump, we first use a binary classifier for distinguishing between concepts and entities, as described in (Cucerzan, 2012). We then use the infobox types as well as specific Wikipedia templates, such as {{person}} and {{disambig}}, to label with a set of 14 entity types (including Common and Disambiguation) a subset of the Wikipedia entries/pages. We further propagate these types to a large fraction of the entities obtained from the Wikipedia dump (2.89 million out of 4.37 million total entries), while allowing multiple labels per entity. Distributions over the entity types are then estimated by employing from the text of the Wikipedia collection the immediate context surrounding interlinks to the pages labeled in the previous step. To limit the noise, we only employ interlinks for which the anchor text is identical with the title of the page linked (also referred to as canonical name).

In the disambiguation stage, we compute the match between the set of entity types associated with each candidate entity with the distributions over entity types for the immediate left and right contexts of the targeted surface form, which are estimated as described above.

2.2.3 Entity Triggers

In the 2013 system, we employ again the Wikipedia interlinks to build an additional vectorial representation of entities. As in the previous work (Cucerzan, 2007), we only use the *bidirectional linkage*, i.e. the links between Wikipedia articles for which the opposite-direction links also exist. However, while in the previous work the Wikipedia interlinks were employed to construct

contexts for entities to be matched directly against the text of the input document, the current system uses the interlinking data to construct *trigger vectors*, which are employed to compute a new matching feature through the document model approach.

Specifically, in the processing of the Wikipedia dump, we associate with an entity E the trigger set composed of all entities F with the property that the Wikipedia page for entity E links to the page for entity F and the page for entity F contains a link to the page for entity E. In the disambiguation stage, we construct the document model component for triggers by aggregating the identifiers for all possible disambiguations of all surface forms in the document. The latent trigger feature is then computed for each candidate entity disambiguation by measuring the similarity of its associated trigger vector and the document model trigger component. As presented in (Cucerzan, 2007), when computing the similarity value, we subtract from the document model the contribution of the surface form being disambiguated.

2.3 Training and Resources

The 2013 MSR system for TAC employs as reference collection the Wikipedia dump from August 5, 2013. The entities from this dump get mapped to the TAC reference collection (which is derived from a 2008 dump) by using the Wikipedia logs for page title changes. No other external data were employed; also, the system does not perform queries against Web data. Both learners employed by the disambiguation stage were trained by using a collection of paragraphs from Wikipedia, in which the interlinks created by the Wikipedia contributors were seen as labeled examples. Additionally, to each paragraph extracted from a page about an entity X we added a standard short sentence of the type "This text is about $[[X|\_SF(X)]]$", where $\_SF(X)$ is a surface form known to have a possible mapping to the entity X (an actual example of training fragment is shown in Figure 1. During training, the system was allowed to extract its own surface forms, and output for those surfaces that matched the Wikipedia links (and only for those) the feature values for all candidate entity disambiguations.

For training the logistic regression learner, we paired the correct disambiguation (as produced by the Wikipedia contributors through interlinking)

> This text is about [[Battle of Waterloo|Waterloo]]. Allegedly, Napoleon tried to escape to North America, but the [[Royal Navy|Royal Navy]] was blockading French ports to forestall such a move. He finally surrendered to [[Captain (Royal Navy)| Captain]] [[Frederick Lewis Maitland (Royal Navy officer)|Frederick Maitland]] of [[Her Majesty's Ship|HMS]] "[[HMS Bellerophon (1786)|Bellero-phon]]" on 15 July. There was a campaign against French fortresses that still held out; [[Longwy|Longwy]] capitulated on 13 September 1815, the last to do so. The [[Treaty of Paris (1815)|Treaty of Paris]] was signed on 20 November 1815. [[Louis XVIII of France|Louis XVIII]] was restored to the throne of France, and Napoleon was exiled to [[Saint Helena|Saint Helena]], where he died in 1821.

Figure 1: Example training fragment, as extracted from the Wikipedia collection and augmented with a first sentence that mentions a surface form of the entity from whose page the Wikipedia text was extracted.

with every of the other candidate entities produced by our disambiguation system; we computed the feature weights by training a binary classifier that employs the feature value differences from each pair. In total, this training process made use of two million such labeled examples.

The boosted-tree ranker was trained using the Microsoft-internal Fast Rank implementation in the TLC library. For each surface form, we labeled the *correct disambiguation* with 1 and all other candidate disambiguations produced by the system with 0, and provided the set of candidates to the trainer. By doing so, we implicitly paired again the correct disambiguation with every other candidate disambiguation. This training process was done on four million training examples.[1] We first performed a parameter sweep by using a subset of the training data, and then we trained the system's ranker by employing the full training set and the following parameters, as obtained from the initial sweep: maximum number of leaves in trees $J = 8$, minimum number of examples in a leaf $L = 40$, and maximum number of trees $N = 200$.

In contrast to the strategies previously used for training the MSR system for the TAC evaluation, the 2013 system does not employ any of the LDC/TAC labeled datasets for training.

[1] The generation of feature values for the Wikipedia training data was done during the night of the evaluation (once the feature computation code was in the final form); because of some unforeseen IT issues, we had to settle for training each learner by using as much data as was produced at the time of generating the two TAC 2013 runs.

2.4 Limiting the Number of Candidates

The system hypothesizes all surface forms (mentions of entities) in a target document and resolves the subset of known surface forms (those extracted/generated from the employed Wikipedia dump). For each surface form, either simple or *composite* (when multiple known surface forms can be extracted at a certain location in text), the system ranks the possible disambiguations, as generated from the Wikipedia dump, in the context of the target document. The average number of disambiguations for each surface form was 41 in the training sets derived from Wikipedia.

2.5 Clustering of Unknown Entities

Similarly to TAC 2012, the MSR system for TAC 2013 does not employ a sophisticated clustering component for NIL-mapped target names. Its NIL-tag labeling relies on the following:

- the much larger size of the 2013 Wikipedia dump (e.g., the target name `Appleton` gets disambiguated by the system to "Appleton, Wisconsin" in two documents and to "Appleton, New York" in two other documents from the TAC 2011 evaluation set, depending on the context of those documents; while both entities appear in the 2013 dumps, only the former is listed in the 2008 reference entity list; however both instances of the latter get assigned the same NIL label despite the fact that the surface forms extracted from text are different from each other: `Appleton, New York` in one instance, and `Appleton, N.Y.` in the other instance);

- acronym expansion matching in the text (e.g., `CASA` gets expanded to "Civil Aviation Safety Authority" in several different documents in the TAC 2011 evaluation set, and thus, it gets mapped to the same NIL identifier; ADF gets mapped to `Alliance Defense Fund` in one document, `American Dance Festival` in another document, and `Australian Defence Force` in a third document, and are assigned identifiers different from each other);

- the identity of surface forms extracted in different target documents by the boundary detection and in-document coreference components (e.g., the target name `Harpootlian` gets mapped to the surface form `Dick Harpootlian` in several documents in the TAC 2011 test set; while there exists no Wikipedia page for this person entity even in the newer collections/dumps, all instances in the TAC 2011 set get clustered together because of the identical surface form to which the target name is mapped).

Accuracy	Systems corresponding to the submitted MSR runs		
	Run1	Run2	TAC Eval
TAC 2011 test set	89.3 %	**89.9** %	86.8%
TAC 2012 test set	**80.4** %	79.3 %	76.2%

Table 1: Accuracy results for the two versions of the current MSR system, which generated the runs submitted at TAC 2013, on the sets employed in the TAC 2011 and 2012 evaluations, as well as the best accuracy scores reported in those evaluations.

2.6 Evaluation on Development Data

Table 1 shows the accuracy scores obtained on the TAC 2011 and 2012 test sets by the two versions of the described system that were then employed to generate the two runs submitted to TAC 2013 for evaluation. Both outperformed by a large margin (about 3 absolute percentage points) the previous official versions of the MSR system. However, no version was consistently better than the other. The version that employs the boosted-trees ranker performed better on the TAC 2011 test set, while the version that employs logistic regression achieved better performance on the TAC 2012 test set. Since the TAC 2013 test data was considered more likely to be sampled from a distribution more similar to that underlying the TAC 2012 data, the logistic-regression-based system was used to generate the MSR run 1 (evaluation id MS_MLI1) and the system that employed boosted trees was used to generate the MSR run 2 (evaluation id MS_MLI2).

2.7 TAC 2013 Entity Linking Evaluation

Table 2 presents the official evaluation results for the two MSR runs, as communicated by NIST. These runs obtained the best overall results in the 2013 evaluation.

Metric	Run1	Run2
B^3+ F1 (Overall −− 2190 queries)	0.720	**0.721**
B^3+ F1 (in KB −− 1090 queries)	0.718	**0.724**
B^3+ F1 (not in KB −− 1100 queries)	**0.720**	0.716
B^3+ F1 (NW docs −− 1134 queries)	0.795	**0.801**
B^3+ F1 (WB docs −− 343 queries)	**0.673**	0.666
B^3+ F1 (DF docs −− 713 queries)	0.623	0.618
B^3+ F1 (PER −− 686 queries)	**0.758**	**0.758**
B^3+ F1 (ORG −− 701 queries)	**0.737**	0.716
B^3+ F1 (GPE −− 803 queries)	0.672	0.693

Table 2: Official results for the two MSR runs submitted for evaluation in TAC 2013. The numbers corresponding to the best performance obtained by any of the par-ticipating systems are bolded.

3 Temporal Slot Filling

3.1 Introduction

Previous work (Agichtein and Gravano, 2000; Etzioni et al., 2004) on relation extraction have targeted the extraction of entity tuples (*e.g.* $presidentOf(BillClinton, USA)$) in order to build a large knowledge base of facts. However, they did not attempt to the capture the temporal validity of the relations extracted. The TAC TSF focuses on this relatively less explored problem: attaching temporal information to relation between entities: $presidentOf(BillClinton, USA)$ is true between the time-frame $1993 - 2001$. The input to a TSF system is a binary relation $spouse(BradPitt, JenniferAniston)$ and a document supporting the relation. The output of the system should be a 4-tuple timestamp [T1, T2, T3, T4] where T1 and T2 are normalized dates denoting the boundaries of the start of the relation and T3 and T4, denoting the end of the relationship. The systems should also provide the offsets of the dates in a supporting sentence within a document *e.g. Pitt married Jennifer Aniston on July 29, 2000 the couple divorced five years later in October 2, 2005*. The 4-tuple normalized timestamp extracted from this example text is [2000-07-29, nil, nil, 2005-10-02]. More details about the task are provided by (Dang and Surdeanu, 2013).

The remainder of this paper describes the participation of the MSR system for Temporal Slot Filling at TAC 2013. For every relation type, our system first automatically extracts the 4-tuple timestamps from each sentence in the source document. Then it uses a language model which is trained on Wikipedia sentences for the relation to classify the top-sentence that supports the given relation between the query entity and the slot filler.

The TSF task for 2013 targeted the following seven relations:

- per:spouse
- per:title
- per:employee_or_member_of
- per:cities_of_residence
- per:statesorprovinces_of_residence
- per:countries_of_residence
- org:top_employees/members

Unfortunately, the training data available for the 2013 evaluation was composed of only 7 examples, one example per relation. Since manual labeling of temporal expressions is expensive, following (Artiles et al., 2011), we built our own training data using distant supervision comprising of a language model from Wikipedia sentences.

3.2 The Temporal Slot Filling Task: Input and Output

The input format for a TSF system is shown below:

- Column 1: query id
- Column 2: slot name
- Column 3: query entity name (subject of relation)
- Column 4: a single docid that justifies the relation between the query entity and the slot filler
- Column 5: a slot filler (possibly normalized, e.g., for dates)
- Column 6: start-end offsets for representative mentions used to extract/normalize the slot filler
- Column 7: start-end offsets for representative mentions used to extract/normalize query entity
- Column 8: start-end offsets of clause(s)/sentence(s) in justification of the relation
- Column 9: confidence score (set to 1.0 for this task)
- Column 10: entitykbid
- Column 11: fillerkbid

Column 1 contains a unique query ID for the relation. Column 3 contains the name of the entity, i.e., the subject of the relation described in this query. Columns 10 and 11 contain the IDs in the KBP knowledge base of the entity and filler, respectively. The rest of the columns are identical to the output of the regular slot filling task, along with Column 4 which contains a valid document ID. Column 9 contains a confidence score set to 1.0 indicating that the relation between the entity is correct. All of the above are provided by TAC. For example, an input query for the relation per:spouse(Brad Pitt, Jennifer Aniston) is:

- Column 1: TEMP72211
- Column 2: per:spouse
- Column 3: Brad Pitt
- Column 4: AFP_ENG_20081208.0592
- Column 5: Jennifer Aniston
- Column 6: 1098
- Column 7: 1492
- Column 8: 1311
- Column 9: 1.0
- Column 10: E0566375
- Column 11: E0082980

For the above query, participating systems will have to extract the temporal validity of the per:spouse relation between the entity Brad Pitt and the slot filler Jennifer Aniston. The output format of a TSF system should contain the following tab-separated values:

- Column 1: query id
- Column 2: the slot name
- Column 3: a unique run id for the submission
- Column 4: one of the strings NIL, T1, T2, T3, or T4
- Column 5: a document ID
- Column 6: start-end offsets for representative mentions used to match/normalize filler
- Column 7: start-end offsets for representative mentions used to match/normalize entity
- Column 8: start-end offsets of clause(s)/sentence(s) in justification for the relation between entity and filler
- Column 9: a normalized date. This date might include Xs if the information is not fully specified in the document, and must include two hyphens to separate month from year and day from month.
- Column 10: provenance of the temporal information

3.3 System Overview

Our target is to use distant supervision from Wikipedia data to build an automatic temporal scoping system. However, for most relations, we find that Wikipedia does not indicate specific start or end dates in a structured form. In addition to this, we need our system to be able to predict whether two entities are currently in a relationship or not based on the document date as well. Hence, in our first step, we build an automatic system which takes as input a binary relation between two entities *e.g.* `per:spouse`(Brad Pitt, Jennifer Aniston) and a number of documents. The system needs to extract highly ranked/relevant sentences, which indicate that the two entities are in the targeted relationship. The next component takes as input the top k sentences generated in the previous step and extracts temporal labels for the input relation. Note that our objective is to develop algorithms that are not relation-specific but rather can work well for a multitude of relations. We elaborate on these two system components further. Figure 2 shows how sentences from Wikipedia can be used to train a system for the temporal slot filling task.

3.3.1 Extracting Relevant Sentences

For every relation, we extract slot-filler names from infoboxes of each Wikipedia article. We also leverage Wikipedia's rich interlinking model to automatically retrieve labeled entity mentions in text. Because the format of the text values provided by different users for the infobox attributes can vary greatly, we rely on regular expressions to extract slot-filler names from the infoboxes. For every relation targeted, we build a large set of regular expressions to extract entity names and filter out noise *e.g.* html tags, redundant text *etc.*.

To extract all occurrences of named-entities in the Wikipedia text, we relabel each Wikipedia article with Wikipedia interlinks by using the MSR EL system. To resolve temporal mentions, we use the Stanford SUTime (Chang and Manning, 2012) *temporal tagger*. Example, for a document published on 2011-07-05, SUTime resolves "last Thursday" to 2011-06-30. It provides temporal tags in the following labels: Time, Duration, Set and Interval. For our experiments we used Time and Duration.

After running the Stanford SUTime, which automatically converts date expressions to their normalized form, we collect sets of contiguous sentences from the page that contain one mention of the targeted entity and one mention of the slot-filler, as extracted by the entity linking system. We then build a large language model by bootstrapping textual patterns supporting the relations, similar to (Agichtein and Gravano, 2000). The general intuition is that a set of sentences that mention the two entities are likely to state something about relationships in which they are.

For assigning a relevance score to sentences with respect to a targeted relation, we represent the sentences in an input document (i.e., Wikipedia page) as d dimensional feature vectors, which incorporate statistics about how relevant sentences are to the relation between a query_entity q and the slot_filler z. For example, for the `per:spouse` relation, one binary feature is "does the input sentence contain the n-gram "QUERY_ENTITY got married"". Note that the various surface forms/mentions of q and z are resolved to their canonical target at this stage.

We were able to extract 61,872 tuples of query entity and slot filler relations from Wikipedia for the `per:spouse` relation. Figure 2 shows how we extract relevant sentences using slot-filler names from Wikipedia. Consider the following text (already processed by our EL system and Stanford SUTime) taken from the Wikipedia page of Tom Cruise:

> On [November 18, 2006$|_{2006-11-18}$], [Holmes$|_{Katie_Holmes}$] and [Cruise$|_{Tom_Cruise}$] were married in [Bracciano$|_{Bracciano}$] ...
> On [June 29, 2012$|_{2012-06-29}$], [Holmes$|_{Katie_Holmes}$] filed for divorce from [Cruise$|_{Tom_Cruise}$] after five and a half years of marriage.

Considering Tom Cruise as the query entity and his wife Katie Holmes as the slot filler for the `per:spouse` relation, we normalize the above text to the following form to extract features:

> On _DATE, SLOT_FILLER and QUERY_ENTITY were married in _LOCATION ...
> On _DATE, SLOT_FILLER filed for divorce from QUERY_ENTITY after five and a half years of marriage.

Our language model consists of n-grams ($n \leq 5$) like "SLOT_FILLER and QUERY_ENTITY were married", "SLOT_FILLER filed for divorce from" which provides clues for the marriage relation. These n-grams are then used as features with an implementation of a gradient boosted decision trees classifier similar to that described by (Friedman, 2001; Burges, 2010). We also use features provided by the EL system which are based on entity types and categories. We call this "relationship" classifier RELCL. The output of this step is a ranked list of sentences which indicate whether there exists a relationship between the query entity and the slot filler.

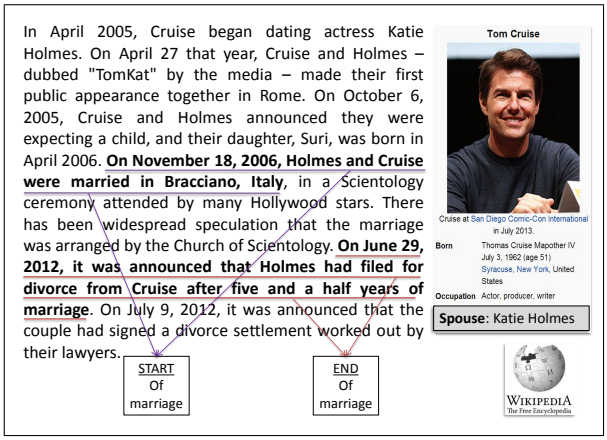

Figure 2: Example of relevant sentences extracted by using query entity and slot-filler names from Wikipedia for the `per:spouse` relation.

3.3.2 Learning Algorithm

Our objective is to rank the sentences in a document based on the premise that entities q and z are in the targeted relation r. We tackle this ranking task by using gradient boosted decision trees (GBDT) to learn temporal scope for entity relations. Previous work such as Sil et al. (2011a; 2011b) used SVMs for ranking event preconditions and (Cucerzan, 2012) and (Zhou et al., 2010) employed GBDT for ranking entities. GBDT can achieve high accuracy as they can easily combine features of different scale and missing values. In our experiments, GBDT outperforms both SVMs and MaxEnt models. The parameters for our GBDT model were tuned on a development set sampled from our Wikipedia dump independent from the training set. These parameters include the number of regression trees and the shrinkage factor.

3.3.3 Gathering Relevant Sentences

On the unseen test data, we apply our trained model and obtain a score for each new sentence s

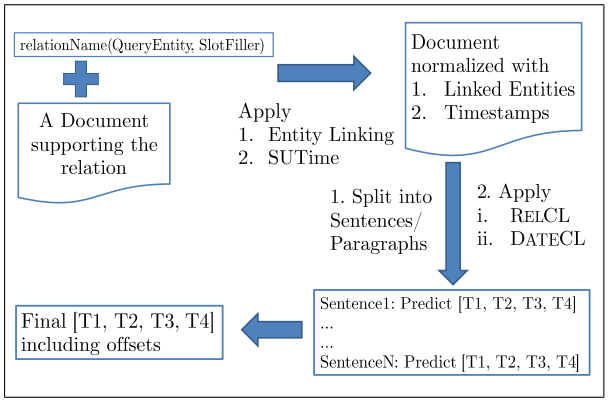

Figure 3: Architecture of the proposed system. Every input document is processed by the (Cucerzan, 2012) entity linking system and the Stanford SUTime system. Temporal information is then extracted automatically using RELCL and DATECL.

that contains mentions of entities q and z that are in a targeted relationship by turning s into a feature vector as shown previously. Among all sentences that contain mentions of q and z, we choose the top k with the highest score.

3.3.4 Extracting Timestamps

To predict timestamps for each relation, we build another classifier, DATECL similar to that described in the previous section, by using language models for "Start", "End" and "In" predictors of relationship. The "Start" model predicts T1, T2; "End" predicts T3, T4 and "In" predicts T2, T3.

Similar to previous work by (Sil et al., 2010) on using discriminative words as features, each of these models compose of "Trigger Words" that indicate when a relationship begins or ends. In the current implementation, these triggers are chosen manually from the language model automatically bootstrapped from Wikipedia. Future directions include how to automatically learn these triggers. For example, for the `per:spouse` relation, the triggers for "Start" contain n-grams such as "married since _DATE" and "married SLOT_FILLER on"; the "End" model contains n-grams such as "estranged husband QUERY_ENTITY", "split in _DATE"; the "In" model contains "happily married", "QUERY_ENTITY with his wife" *etc.*. For an input sentence with query entity q and slot-filler z, a first class of raw trigger features consists of cosine-similarity(Text(q, z), Triggers(r)) where $r \in Start, End, In$. Here, Text(q, z) indicates the full sentence as context. We also employ another feature that computes cosine-similarity(Context(q, z), Triggers(r)), which constructs a *mini-sentence* Context(q, z) from the original by choosing windows of three words before and after q and z, and ignoring duplicates.

The MSR TSF system also considers the presence of other events as triggers *e.g.* a "death" event signaled by "SLOT_FILLER died" might imply that a relationship ended on that timestamp. Similarly, a "birth" event can imply that an entity started living in a particular location *e.g.* the `per:born-In`(Obama, Honolulu) relation from the sentence "President Obama was born in Honolulu in 1961" indicates that T1 = 1961-01-01 and T2 = 1961-12-31 for the relation `per:cities_of_residence`(Obama, Honolulu).

At each step, our system extracts the top timestamps for predicting "Start", "End" and "In" based on the confidence values of DATECL. Similar to previous work by (Artiles et al., 2011), we aggregate and update the extracted timestamps using the following heuristics:

Step 1: Initialize T= [-∞, +∞, -∞,+ ∞]
Step 2: Iterate through the classified timestamps
Step 3: For a new T' aggregate :

$$T \&\& T' = [max(t_1, t'_1), min(t_2, t'_2), \\ max(t_3, t'_3), min(t_4, t'_4)]$$

Update only if: $t_1 \leq t_2$; $t_3 \leq t_4$; $t_1 \leq t_4$

This novel two-step classification strategy removes noise introduced by distant supervision training and decides if the extracted (entity, filler, timestamp) tuples belong to the relation under consideration or not. For example, for the `per:spouse` relation between the entities Brad Pitt and Jennifer Aniston, the MSR TSF system extracts sentences like "*..On November 22, 2001, Pitt made a guest appearance in the television series Friends, playing a man with a grudge against Rachel Green, played by Jennifer Aniston..*" and "*Pitt met Jennifer Aniston in 1998 and married her in a private wedding ceremony in Malibu on July 29, 2000..*". Note that both sentences contain the query entity and the slot filler. The system automatically rejects the extraction of temporal information from the former even though the sentence contains mentions of both entities. This is because the language model for the marriage relation does not match well this candidate sentence, which is actually focussing on the two entities being in the different relation of co-acting/appearing

in the same motion picture. The latter sentence is determined as matching the language model for the marriage relation, and our system extracts the temporal scope July 29, 2000 and attaches the START label to it. Most previous systems do not perform this noise removal step, which is a critical component in our distant supervision approach.

3.4 Evaluation

Our system was trained on the Wikipedia dump of May 2013. We set aside a portion of the dump as our development data. We chose the top-relevant nGrams based on the performance on the development data as features. The final evaluation data to test our system is provided by TAC.

In order for a temporal constraint (T1-T4) to be valid, the document must justify the query relation (similar to the regular English slot filling task) AND the temporal constraint. TAC uses the following metric to evaluate the performance of each system as the time information provided by the texts may be only approximate: score measuring the similarity of each constraint in the key and system response: Suppose that the date in the gold standard be k_i and the date in our system response be r_i. Also, let $d_i = |k_i - r_i|$, measured in years. Then the score for the set of temporal constraints on a slot is:

$$S(slot) = \frac{1}{4}\sum_{i=1}^{4}\frac{c}{c+d_i} \quad (1)$$

$$c = \begin{cases} c_{overconstrained} \\ c_{vagueness} \end{cases}$$

Here, $c_{overconstrained}$ when (i $\in \{1,3\}$ AND $r_i > k_i$) OR (i $\in \{2,4\}$ AND $r_i < k_i$) and $c_{vagueness}$, otherwise. These conditions are set to 1 year such that errors of that amount get 50% credit. Hence, this yields a score between 0 and 1. The absence of a constraint in T1 or T3 is treated as a value of $-\infty$; the absence of a constraint in T2 or T4 is treated as a value of $+\infty$. For more details on the scoring, see (Dang and Surdeanu, 2013).

Table 4 shows our results in the track. Of the 273 evaluation queries that were distributed, only 201 were considered for the purposes of scoring as per TAC judges mainly because of the absence of slot-fillers in some documents and wrong temporal information which violates certain constraints. Our system has obtained the best score (in both of the submitted runs) outperforming all the other systems that participated in this track. Our second run (run: MS_MLI_2) which imposes more constrains in the temporal relation has obtained the highest score. Comparison with a reasonable baseline similar to (Ji et al., 2011) and LDC experts are shown in Table 4. This baseline makes the simple assumption that the corresponding relation is valid at the document date. That means that it creates a "within" tuple as follows: $< -\infty, doc\_date, doc\_date, +\infty >$. Hence, this baseline system for a particular relation always predicts T2 = T3 = the date of the document.

Run ID	Score
MS_MLI1	0.267
MS_MLI2	**0.331**

Table 3: Our system run ids along with their scores. Our second run outperforms the first.

4 Conclusion

The paper described the MSR systems for entity linking and temporal slot filling that participated in the TAC 2013 evaluation. The entity linking system, which uses the same architecture as the MSR systems that participated in the previous two TAC entity linking evaluations, obtained top empirical results on the two runs submitted for the 2013 entity linking evaluation. The temporal slot filling system uses distant supervision from Wikipedia data, which is shown to be an effective resource for addressing the temporal task. The two runs generated by this system obtained the best two scores in the 2013 temporal slot filling evaluation.

Acknowledgments

The authors would like to thank Mikhail Bilenko and Matthew Richardson for their help with integrating the TLC machine learning library, and Andrzej Pastusiak for his help in indexing the provided corpus of documents, as well as other useful discussions.

References

E. Agichtein and L. Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Procs. of the Fifth ACM International Conference on Digital Libraries*.

Javier Artiles, Qi Li, Taylor Cassidy, Suzanne Tamang, and Heng Ji. 2011. CUNY BLENDER

	S1	S2	S3	S4	S5	S6	S7	**ALL**	StDev
Baseline	24.70	17.40	15.18	17.83	14.75	21.08	23.20	19.10	3.60
MS_MLI2	**31.94**	**36.06**	**32.85**	**40.12**	**33.04**	**31.85**	27.35	**33.15**	3.66
LDC	69.87	60.22	58.26	72.27	81.10	54.07	91.18	68.84	12.32

Table 4: Results for the TAC-TSF 2013 test set, overall and for individual slots. The slots notation is: S1: org:top members employees, S2: per:city of residence, S3: per:country of residence, S4: per:employee or member of, S5: per:spouse, S6: per:statesorprovince of residence, S7: per:title. The score for the output created by the LDC experts is also shown.

TACKBP2011 Temporal Slot Filling System Description. In *TAC*.

Chris Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11:23–581.

Angel X Chang and Christopher Manning. 2012. Sutime: A library for recognizing and normalizing time expressions. In *LREC*, pages 3735–3740.

X. Cheng and D. Roth. 2013. Relational inference for wikification. In *Proceedings of EMNLP 2013*, pages 1787–1796.

S. Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of EMNLP-CoNLL 2007*, pages 708–716.

Silviu Cucerzan. 2012. The MSR System for Entity Linking at TAC 2012. In *TAC*.

Hoa Trang Dang and Mihai Surdeanu. 2013. Task description for knowledge-base population at TAC 2013. In *TAC*.

O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. 2004. Web-Scale Information Extraction in KnowItAll. In *WWW*, New York City, New York.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232.

W. Gale, K. Church, and D. Yarowsky. 1992a. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992b. One sense per discourse. In *The 4th DARPA Speech and Natural Language Workshop*, pages 233–237.

A. R. Golding. 1995. A Bayesian hybrid method for contextsensitive spelling correction. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 39–53.

Heng Ji, Ralph Grishman, and Hoa Trang Dang. 2011. Overview of the tac2011 knowledge base population track. In *TAC*.

P. McNamee and H.T. Dang. 2009. Overview of the tac 2009 knowledge base population track. In *Proceedings of the Text Analysis Conference 2009*.

Avirup Sil and Alexander Yates. 2011a. Extracting STRIPS representations of actions and events. In *RANLP*.

Avirup Sil and Alexander Yates. 2011b. Machine Reading between the Lines: A Simple Evaluation Framework for Extracted Knowledge Bases. In *Workshop on Information Extraction and Knowledge Acquisition (IEKA)*.

Avirup Sil, Fei Huang, and Alexander Yates. 2010. Extracting action and event semantics from web text. In *AAAI Fall Symposium on Common-Sense Knowledge (CSK)*.

Yiping Zhou, Lan Nie, Omid Rouhani-Kalleh, Flavian Vasile, and Scott Gaffney. 2010. Resolving surface forms to wikipedia topics. In *COLING*, pages 1335–1343.