Learning Constraints for Consistent Timeline Extraction

David McClosky and Christopher D. Manning
Natural Language Processing Group
Computer Science Department
Stanford University, Stanford, CA, USA
{mcclosky,manning}@stanford.edu

Abstract

We present a distantly supervised system for extracting the temporal bounds of *fluents* (relations which only hold during certain times, such as *attends school*). Unlike previous pipelined approaches, our model does not assume independence between each fluent or even between named entities with known connections (parent, spouse, employer, etc.). Instead, we model what makes timelines of fluents consistent by learning cross-fluent constraints, potentially spanning entities as well. For example, our model learns that someone is unlikely to start a job at age two or to marry someone who hasn't been born yet. Our system achieves a 36% error reduction over a pipelined baseline.

1 Introduction

Many information extraction (IE) systems traditionally extracted just relations, but a great many real world relations such as *attends school* or *has spouse* vary over time. To capture this, some recent IE systems have extended their focus from relations to *fluents* (relations combined with temporal bounds). This can be seen in the temporal slot filling track in the TAC-KBP 2011 shared task (Ji et al., 2011). A direct application of this work is the automatic improvement of online resources such as Freebase and Wikipedia infoboxes. Indirect applications include question answering systems.

Fluents can be grouped together to form timelines (see Figure 1 for an example) and provide easily capturable consistency constraints. Our goal is to learn these constraints and use them to produce more accurate timelines of significant events for people and organizations. For example, it is common knowledge that someone cannot attend a school if they haven't been born yet. Constraints on consistent timelines do not need to be hard constraints, though: it is rare, although possible, to become the CEO of a company at the age of 21.

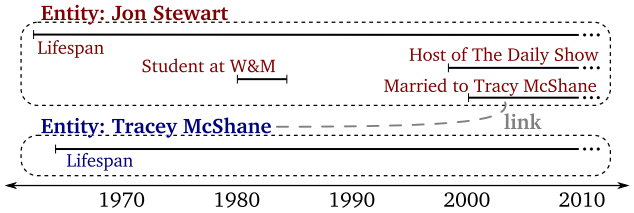

Figure 1: A timeline of two named entities. Each time span represents a *fluent* (a relation with temporal bounds). Temporal bounds are denoted by spans on the timeline. Fluents can create links between entities (e.g., marriage).

Despite the rich constraints on valid timelines, there is relatively little work on exploiting these constraints for mutual disambiguation. Many existing systems extract different parts of a timeline separately and use heuristics to combine them. These heuristics tend to optimize only local consistency (within a single fluent) but ignore more global constraints across fluents (e.g., attending a school before being born) or across fluents of two linked entities (e.g., attending a school before the school was founded). In this work, we explore using joint inference to enforce these constraints. We show that these techniques can yield substantial improvements. Additionally, our general approach is not specific to extracting temporal boundaries of fluents. It could easily be applied to other IE systems which

employ independent extractions followed by heuristics to improve consistency.

2 The timelining task

As a basis for our task, we first describe the Temporal KBP task (Ji et al., 2011). As input, one is given a list of *queries*, a database of example fluents, and source documents. Queries are named entities (people or organizations) with their gold relations but no temporal bounds. The database consists of training entities with their fluents, including known temporal bounds for each fluent. Example fluents can be seen in Table 1. Note that the database may be incomplete. In addition to missing fluents for an entity, some temporal bounds may be missing from the database; missing bounds are unfortunately indistinguishable from unbounded ranges. As a result, we can only trust concrete temporal boundaries in the database. Source documents consist of raw text from news, blogs, and Wikipedia articles. For each fluent, systems must output their predicted temporal bounds, along with references to source documents to provide provenance.

Our task is a variation of the Temporal KBP task. In our case, the database is a collection of Freebase[1] entities and their fluents, merged with Wikipedia infoboxes. Each entity has a unique ID, allowing us to avoid some coreference issues (though there can still be issues in document retrieval). In Temporal KBP, the temporal representation allows for upper and lower bounds on both the event start and end: $\langle s_l, s_u, e_l, e_u \rangle$ where $s_l \leq start \leq s_u$, $e_l \leq end \leq e_u$. However, it is difficult to obtain these bounds without manual annotation. As a result, we opted for the simpler representation which can be easily found in databases like Freebase. Our temporal representation is limited to bounds of the form $\langle start, end \rangle$ where either can be unbounded or unknown (both represented as $\pm\infty$).

Our set of fluents is closely related to those in the Temporal KBP task. Our goal was to use as much temporal information as possible, with the hope of each fluent providing additional potential constraints. While we omit the *resides in* and *member of* fluents,[2] we add several others. For people and organizations, we add a special fluent, *lifespan*, which doesn't take a slot value.[3] A list of fluents we use are listed in Table 3.

3 Model

To operate on a set of queries, we first collect candidate temporal expression mentions for each fluent from our source documents. This limits us to using temporal expression mentions which appear near fluent mentions in text. It also ensures that we can provide provenance for each temporal boundary assertion. This process is described in §3.1.

Our model contains two components, both of which assign probabilities to timelines. The *classifier component* determines how each candidate temporal expression mention connects to its fluent (§3.2). For example, the mention may indicate the START of the fluent, the END, both its START AND END (for instantaneous events), or be UNRELATED. These connections involve relations between temporal expression mentions and relations and we refer to them as *metarelations*.[4] For features, the classifier uses the surrounding textual and syntactic context of temporal expression and fluent mentions. Each classification decision is made independently, allowing for inconsistency at multiple levels (within a fluent, across fluents, or across entities). However, using joint inference, the classifier component can determine the best overall span for each fluent.

The *consistency component* learns what makes timelines consistent (§3.3). It is similar in nature to a language model for timelines instead of sentences. Given a candidate timeline, the consistency component estimates its probability of occurring. This is done by decomposing timelines into a series of *questions* (such as "did the entity go to school before starting a job?") and learning the probabilities of different answers from training data.

Unlike the classifier component, the consistency component is blind to the underlying text in the source documents. The two components work together to find a global timeline that is both based on textual evidence and coherent across entities using

[1] `http://freebase.org`

[2] This is because these fluents are rarely present in Freebase with temporal bounds.

[3] Note that this is a relation in the non-temporal KBP task.

[4] Other metarelations are possible under more complex temporal representations. For example, Artiles et al. (2011) uses the HOLDS metarelation.

Entity	Relation	Slot value	Temporal bounds
Jon Stewart `/en/jon_stewart`	*lifespan*	—	[1962-11-28, $+\infty$)
Jon Stewart `/en/jon_stewart`	*has parent*	Donald Leibowitz `/en/donald_leibowitz`	[1962-11-28, $+\infty$)
Jon Stewart `/en/jon_stewart`	*attends school*	College of William and Mary `/en/college_of_william_and_mary`	($-\infty$, 1984]
Jon Stewart `/en/jon_stewart`	*has spouse*	Tracey McShane `/en/tracy_mcshane`	[2000-11, $+\infty$)

Table 1: Example relations with their temporal bounds. Freebase IDs are shown in `monospace`. Note that temporal bounds differ in their resolution (some are days of the year, others are only years). Some bounds are unknown (e.g., the start of the *attends school* fluent) and indistinguishable from unbounded. The *lifespan* fluent is a unary relation.

joint inference (note that they are trained independently). The inference process is described in §3.4.

3.1 Temporal expression retrieval

Given a fluent, we search for all textual mentions of the fluent and collect nearby temporal expression mentions. These temporal expressions are used as candidate boundaries for the fluent in later steps. The search process assumes that if a fluent's entity and slot value co-occur in a sentence,[5] that sentence is typically a positive example of the fluent.[6] This is sometimes known as *distant supervision* (Craven and Kumlien, 1999; Mintz et al., 2009). We use the Stanford Core NLP suite (Toutanova et al., 2003; Finkel et al., 2005; Klein and Manning, 2003; Lee et al., 2011) to annotate each document with POS and NER tags, parse trees, and coreference chains. On top of this, we apply a rule-based temporal expression extractor (Chang and Manning, 2012). Since we have coreference links, we also search documents for anything coreferent with the fluent's entity.

The temporal expression extractor handles most standard date and time formats. For each document, one can provide an optional reference time. For underspecified dates, the reference time is used to resolve these dates to full expressions if possible. Some of our documents are news articles, where we use the publication date as the reference time. Other documents, e.g., Wikipedia articles, are undated and we typically omit a reference time for these. We exclude dates which are not uniquely resolvable (e.g., "September 15th," when the reference date is unknown) since our task requires us to output unambiguous dates.

We create training datums by computing the metarelation between each temporal expression and its gold fluent. For example, for the temporal expression mention "September 15th, 1981" and gold *lifespan* relation that spans [`1981-09-15`, $+\infty$), we would assign the START metarelation. As a heuristic, we allow for underspecified matches. Thus, both "1981" and "September 1981" would have the START metarelation but "September 2nd, 1981" would be assigned UNRELATED.

3.2 Classifier component

We use a classifier to determine the nature of the link between fluents and candidate temporal expression mentions. Our classifier (a standard multiclass maximum entropy classifier) learns a function $C : (t, f) \rightarrow \mathcal{M}$ where t is a temporal expression mention, $f = \langle$entity, relation name, slot value$\rangle$ is a fluent from the database, and $\mathcal{M}$ is the set of the four possible metarelations.

Features for the classifier include many of those in Artiles et al. (2011). These include standard relation extraction features such as the dependency paths between the temporal expression and the entity or slot value. We use both the original depen-

[5] While we limit our scope to sentences in this work, it is trivial to extend this to larger regions such as paragraphs.

[6] The *lifespan* fluent requires special handling. Ideally, its candidates would be provided by a relation extraction mention detector (e.g., a KBP system). For this work, we use the gold *lifespan* bounds as slot values for the purpose of document retrieval. While this does heavily bias the system towards using gold bounds, the system still must predict the correct associations (START, END, etc.) making the *lifespan* fluent non-trivial.

dency paths and their collapsed Stanford Dependencies forms (de Marneffe and Manning, 2008). We include the lengths of each path and, if the path is shorter than four edges, the grammatical relations, words, POS tags, and NER labels along the path. We extract the same sorts of features from surface paths (i.e., the words and tags between the entity and the temporal expression) if the path is five tokens or shorter. For temporal expressions, we include their century and decade as features. These features act as a crude prior over when valid temporal expressions occur. There are also features for the precision of the temporal expression (year only, has month, and has day). Lastly, we include the relation name itself as a feature.

Previous work (Artiles et al., 2011) heuristically aggregates the hard decisions from their classifier to create a locally consistent span. The *basic aggregation model* (described in §4.2) is similar to their method. In contrast, our method uses the likelihood of complete spans to ensure both boundaries are consistent with the text.

To calculate the likelihood of a specific temporal span for a fluent f, we represent the span as a series of metarelations and take the product of their probabilities. For example, if the candidate span is [`1981-09-15`, $+\infty$) and we have two temporal expressions, "September 15th, 1981" and "2012":

$$P\big(span(f) = [\texttt{1981-09-15}, +\infty) \mid f\big) = P\big(C(\text{“September 15th, 1981”}, f) = \textsc{start}\big) \times P\big(C(\text{“2012”}, f) = \textsc{unrelated}\big)$$

This can easily be extended to calculating the joint probability of an entire timeline, represented as a list of $\langle \textit{fluent}, span \rangle$ pairs:

$$P_{CC}\big(\langle f_1, s_1 \rangle, \dots\big) = \prod_i P\big(span(f_i) = s_i \mid f_i\big)$$

We refer to this model as the Combined Classifier (CC) since it uses the probabilities of all timelines boundaries rather than aggregating hard local decisions.

3.3 Consistency component

While distant supervision can be used to create implicit negative examples for the classifier component (time expressions marked as UNRELATED), we do not have an equivalent technique to reliably create negative examples for the consistency component (examples of inconsistent timelines). Instead, we only have positive examples of consistent timelines from the database. As a result, we must treat predicting consistency as a density estimation rather than a classification problem.

Our consistency component is designed to be as general as possible – it does not even include basic constraints about timelines such as "starts are before ends." Instead, we provide several simple templates for temporal constraints to allow it to learn these basic tendencies as well as more complex ones. Examples include whether one typically goes to school first or starts their first job, how many jobs people typically have at one time, or if it is possible to marry someone who hasn't been born yet.

We achieve this by decomposing timelines into a series of probabilistic events, or *questions*. As an example, one question about the timeline shown in Table 1 is whether Jon Stewart graduated from the College of William and Mary BEFORE marrying Tracey McShane, i.e., *end*(*attends school*) $<$ *start*(*has spouse*). In this case, the answer is "yes." More generally, we can apply the BEFORE template to all boundaries of all fluents: $\textit{boundary}_1(\textit{fluent}_1) < \textit{boundary}_2(\textit{fluent}_2)$. We use templates like these (denoted by SMALL CAPS) to generate all possible questions to ask about a specific entity.

Other questions can be asked at the fluent level rather than the boundary level (Allen, 1983). One set of fluent level questions asks whether two fluents' spans OVERLAP. For example, in Table 1, Jon Stewart's *lifespan* OVERLAPS with the span of his *has spouse* fluent. Other sets of fluent level questions ask whether the span of a fluent completely CONTAINS the span of another one, whether a fluent is COMPLETELY BEFORE another fluent, and whether two fluents TOUCH (the start of one fluent is the same as the end of another).

Since all of these questions involve ordering but ignore the actual differences in time, we create one more set of questions asking whether two boundaries are WITHIN a certain number of years:

$$|\textit{boundary}_1(\textit{fluent}_1) - \textit{boundary}_2(\textit{fluent}_2)| \leq K$$

for $K \in \{1, 2, 4, 8, 16\}$. The aim is to approximate the typical lengths of a single fluent or amount of time between boundaries from different fluents.

There is nothing which requires that the fluents in question come from a single entity. Thus, we can trivially ask questions about two entities which are linked by a fluent. For example, since Jon Stewart is linked to Tracey McShane by the *has spouse* fluent (Table 1), we could ask the question of whether Jon Stewart's *lifespan* OVERLAPS Tracey McShane's *lifespan*. We can ask any type of question about two linked entities and distinguish the questions by prefixing them with the nature of the link (*has spouse* in this case).

Note that not all questions can be answered since they may rely on comparing unknown values. This is because (for our setup) infinite values are indistinguishable from unknown values. For example, the start of the Jon Stewart's *attends school* fluent is undefined in the database, but clearly not actually $-\infty$. Thus, we add a third possible answer to each question: unknown. The answers to boundary level questions are defined only if both boundaries are finite. Fluent level questions have known answers as long as both fluents have at least one finite value.

To train our model, we gather the answers to questions over all the fluents from training entities. Each question forms a multinomial over the three possible values (yes, no, unknown). To determine the probability of a complete timeline:

$$P_{consistency}(timeline) = \prod_{(q,a) \in Q(timeline)} \begin{cases} (1-c)P_\theta(a \mid q) & q \text{ is old} \\ c & q \text{ is new} \end{cases}$$

where $Q(\cdot)$ generates all possible $\langle question, answer \rangle$ pairs which are consistent with the fluents in the timeline, θ is a vector of the model parameters, and c is a smoothing parameter (described below).

To learn the model parameters, we start by using maximum-likelihood estimation for these multinomials from training entities. However, some smoothing is required since new entities may contain previously unseen answers to existing questions. To address this, we apply add-λ smoothing to each multinomial, $P_\theta(a \mid q)$. Additionally, it is possible to see entirely new questions when we see a new combination of fluent types. We reserve an amount of probability mass for new questions, c. c and λ are estimated in turn by picking the value that maximizes the likelihood of the timeline made by the development entities.

To adjust the weight of the consistency component relative to the classifier component, we take the geometric mean of the likelihood using the total number of questions, $|Q(t)|$, as the exponent and raise the resulting mean to an exponent, β. This is necessary since the two components essentially operate on different scales. The Joint Classifier with Consistency (JCC) model calculates the score of a timeline, t, according to both components:

$$\text{score}_{JCC}(t) = P_{CC}(t) \left[P_{consistency}(t)^{\frac{\beta}{|Q(t)|}} \right]$$

3.4 Inference

Inference for the CC model is relatively simple: Simply pick the most likely span for each fluent. Since it assumes all fluents are independent, the bounds for each fluent can be inferred separately. To perform inference on a specific fluent, we consider all of its possible temporal spans, limited by the temporal expression mentions found by the retrieval system (§3.1). Each possible span assigns one of the four metarelations to each candidate temporal expression for the fluent. For example, if we found only the temporal expression mention "1981" for a specific fluent, there are four possible spans:

UNRELATED:	$(-\infty, +\infty)$
START:	[1981-01-01, $+\infty$)
END:	$(-\infty$, 1981-12-31]
START AND END:	[1981-01-01, 1981-12-31]

Note that when we assign "1981" as a start, we use the earliest possible time (January 1st) while when we assign it as an end, we use the latest possible time (December 31st). Of course, we typically have multiple candidate temporal expressions and thus potentially many more than four possible spans. All temporal expression mentions that resolve to the same time are grouped together, since it wouldn't make sense to assign "August 28th, 2010" one metarelation and a different one to "8/28/2010."

Joint inference for the JCC model is a little more involved since the consistency model does not assume independence across fluents. Thus, we need

to apply techniques like Gibbs sampling or random-restart hillclimbing (RRHC) to determine the optimal temporal spans for each fluent. For our task, the two methods obtain similar performance while RRHC requires many fewer iterations so our discussion focuses on the latter. RRHC involves looping over all fluents in our testing entities, shuffling the order of the fluents at the beginning of each pass. We maintain a working timeline, t, with our current guesses of the spans for each fluent. For each fluent and span $\langle f, s\rangle \in t$, we pick the optimal span for f:

$$s^* = \arg\max_{s' \in S(f)} \text{score}_{JCC}(t_{s'})$$

where $S(f)$ determines all possible temporal spans for the fluent f and $t_{s'} = (t \cup \langle f, s'\rangle) - \langle f, s\rangle$ is a copy of t where s' is the span for f instead of s. After selecting s^*, we add it to our timeline: $t_{new} = (t \cup \langle f, s^*\rangle) - \langle f, s\rangle$. Rather than calculating the score of the full timeline, we can save time by using only the relevant fluents in $t_{s'}$. For example, if our fluent is the *has spouse* fluent for Jon Stewart, we include all the fluents involving Jon Stewart and any relevant linked entities. In this case, we also include all the fluents for Tracey McShane.

Each round of RRHC consists of two passes through the fluents we are inferring: An $\arg\max$ pass followed by a randomization pass where we randomly choose spans for a random fraction of the fluents. When finished, we return the highest scoring timeline seen during either of these passes.

4 Experiments

We evaluate our models (CC and JCC) according to their ability to predict the temporal bounds of fluents from Freebase. This is similar to the Diagnostic Track in the Temporal KBP task, where gold relations are provided as inputs. We provide three baselines for comparison, discussed further in §4.2. To form our database, we scraped a random sample of people and organization entities from Freebase using their API. Since our consistency model has limited effect if entities do not have any links to other entities, we restrict our attention to entities linked to at least one other entity – this eliminates a large portion of possible entities. Our corpus[7] consists of 8,450 entities for training, 1,072 for development, and 1,067 for test. Entities have approximately 2.0 fluents on average.

From experiments on the development set, we set the relative strength of the consistency component $\beta = 10$. For the JCC model, we perform three runs for each experiment with different random seeds. Each experiment performs 10 rounds of RRHC,[8] initializing from an empty timeline.

4.1 Evaluation metric

Our evaluation metric is adapted from the Temporal KBP metric (Ji et al., 2011) to work with 2-tuples for temporal representations rather than the 4-tuples in Temporal KBP. The metric favors tighter bounds on fluents while giving partial credit. All dates need to be given at day resolution. Thus, for gold fluents with only year- or month-level resolution, we treat them as their earliest (for starts) or latest (for ends) possible day. To score a boundary, we take the difference between the predicted and gold values: If they're both unbounded ($\pm\infty$), the boundary's score is 1. If only one is unbounded, the score is 0. If both are finite, the score is $1/(1 + |d|)$ where d is the difference between the values in years. To score a fluent, we average the scores of its start and end boundaries. In rare cases, we have multiple spans for the same relation (e.g., Elizabeth Taylor married Richard Burton twice). In these cases, we give systems the benefit of the doubt and greedily align fluents in such a way as to maximize the metric. The total metric computes the score of each fluent divided by the number of fluents. The official metric includes precision and recall components, but since our setup provides gold relations, our precision and recall are be equal. This allows us to report a single number.

4.2 Baselines and oracle

The simplest baseline is the *null* baseline, proposed in Surdeanu et al. (2011). This baseline assumes that all fluents are unbounded in their spans. The purpose

[7] `http://nlp.stanford.edu/~mcclosky/data/freebase-temporal-relations.tar.gz`

[8] There was no significant difference in accuracy between running 10 and 200 rounds of RRHC.

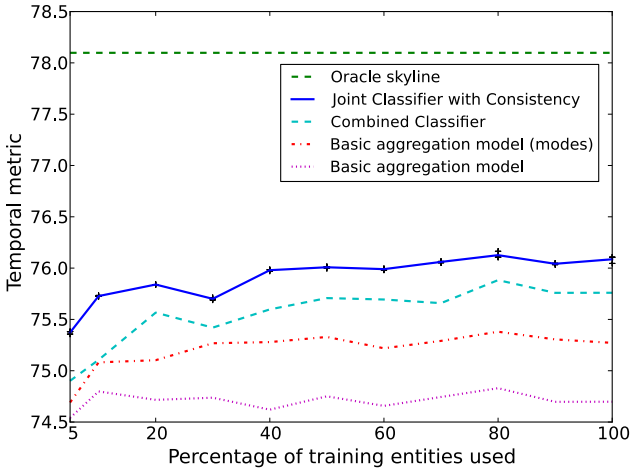

Figure 2: Performance of models and baselines on development data while varying amount of training data. Not pictured: The *null* baseline at 58.8%.

of this baseline is primarily to show the approximate minimal value for the temporal metric.

We provide two other baselines to describe heuristic methods of aggregating the hard decisions from the classifier function C learned in §3.2. These are unlike the CC model which uses the soft decisions of C. Both of these baselines maintain lists of possible starts and ends for each fluent. If the classifier assigns START AND END, we add the candidate temporal expression to both. The first baseline, *basic aggregation*, is along the same lines as the aggregation method used in Artiles et al. (2011), a state-of-the-art system. Our baseline assigns the earliest start and the latest end as the bounds for each fluent, assigning $\pm\infty$ for empty lists. The second baseline, *basic aggregation (modes)*, is the same except that it uses the mode from each list.

To determine the best possible score given our temporal expression retrieval system, we calculate the oracle score by assigning each fluent the span which maximizes the temporal metric. The oracle score can differ from a perfect score since we can only use candidate temporal expressions as values for a fluent if (a) mentions of the fluent are retrievable in our source documents, (b) the temporal expression mention appears nearby, and (c) our temporal expression extractor is able to recognize it correctly. Nevertheless, it is still a reasonable upper bound in our setting.

Model	Dev	Test
Oracle	78.1	75.2
Joint Classifier with Consistency	**76.1**	**72.2**
Combined Classifier	75.8	71.5
Basic aggregation (modes)	75.3	71.2
Basic aggregation	74.7	70.5
null baseline	58.8	55.6

Table 2: Performance of systems on development and test divisions. The Joint classifier with Consistency is the average of three runs with negligible variance ($\sigma \approx 0.02$).

4.3 Results

We present the performance of our models, baselines, and the oracle in Figure 2 while varying the percentage of training entities. The JCC model (76.1% on development with 100% training entities) is consistently the best non-oracle system. Its gains are larger when the amount of training data is low. This is presumably because the classifier suffers from insufficient data and the consistency component is able to learn consistency rules to recover from this. Both the CC and JCC models outperform the basic aggregation models. This shows the value of incorporating all marginal probabilities. On the test set (Table 2), the JCC model performs even better in comparison to the simple models, despite the test set being clearly more difficult than the development set. In this case, the JCC achieves a 36% error reduction over the basic aggregation model.[9] On the official KBP entities, the oracle score is 92%. Since we use a different set of entities, there is a mismatch between our entities and the source documents resulting in a lower oracle score. Addressing this is future work.

5 Discussion

Table 3 shows the performance of four systems and baselines on individual fluent types. The JCC model derives most of its improvement from the two *lifespan* fluents and other high frequency fluents. The *lifespan* fluents provide the most room for improvement since they tend to contain non-*null* values a reasonable amount of the time (note how these relations have a large gap between their ora-

[9] This counts errors relative to the oracle score since we treat the retrieval system as fixed in this work.

		Model					
Fluent	Count	*null*	Basic	Basic (modes)	CC	JCC	Oracle
organization: *lifespan*	266	49.2	71.0	70.7	71.1	71.7	73.4
organization: *top employees*	150	88.0	88.0	88.0	88.0	88.0	88.3
organization: *founders*	31	0.0	5.4	5.4	10.8	11.1	16.3
organization: *acquires company*	14	21.4	21.4	21.4	21.4	21.4	38.5
person: *lifespan*	806	28.6	63.1	64.6	65.6	66.1	69.1
person: *has spouse*	582	92.2	92.1	92.1	92.2	92.3	93.1
person: *attends school*	107	97.7	97.7	97.7	97.7	98.1	98.1
person: *has job*	85	78.8	79.4	79.4	78.8	78.8	80.3
person: *holds government position*	45	16.7	19.7	19.7	19.7	19.7	25.1
person: *romantic partner*	5	50.0	52.9	52.9	52.9	52.9	71.2

Table 3: Fluent-level performance of models and baselines on development data. Scores are calculated with the temporal metric. CC stands for Combined Classifier and JCC for Joint Classifier with Consistency. The JCC model obtains most of its benefits on the two *lifespan* relations. For *attends school*, it is the only system able to achieve oracle-level performance. The *null* baseline is especially strong for several fluents since these tend to be unbounded or (more likely) missing their values in Freebase. The two basic aggregation models differ primarily on their predictions for the *lifespan* fluents.

cle and *null* scores). Additionally, the *lifespan* fluent is always present for entities while other fluents are sparser. For *attends school*, JCC is the only system able to achieve oracle-level performance. No system improves on the *null* baseline for *acquires company*. This is likely due to its sparsity.

Inspecting the multinomials in the consistency component, we can see that the model learns reasonable answers to questions such as whether an entity "was born before getting married?" (yes: 14.8%, no: 0.04%),[10] "died before their parents were born?" (yes: 0.3%, no: 53.7%) and "finished a job before starting a job (not necessarily the same one)?" (yes: 72.5%, no: 20.5%). Despite some unavoidable noise in the data, it is clear these constraints are useful.

6 Related work

There is a large body of related work that focuses on ordering events or classifying temporal relations between them (Ling and Weld, 2010; Yoshikawa et al., 2009; Chambers and Jurafsky, 2008; Mani et al., 2006, *inter alia*). Much of this work uses the Allen interval relations (Allen, 1983) which richly describe partial orderings of fluents. We use several of these as fluent-level question templates.

Joint inference has been applied successfully to other NLP problems (Roth and Yih, 2004; Toutanova et al., 2008; Martins et al., 2009; Chang et al., 2010; Koo et al., 2010; Berant et al., 2011). Two recent examples in information extraction include using Markov Logic for temporal ordering (Ling and Weld, 2010) and using dual-decomposition for event extraction (Riedel and McCallum, 2011).

Our work is closest to Temporal KBP slot filling systems. The CUNY and UNED systems (Artiles et al., 2011; Garrido et al., 2011) for this task used classifiers to determine the relation between temporal expressions and fluents. These systems use the hard decisions from the classifier and combine the decisions by finding a span that includes all temporal expressions. In contrast, our system uses the classifier's marginal probabilities along with the consistency component to incorporate global consistency constraints. Other participants used rule-based and pattern matching approaches (Byrne and Dunnion, 2011; Surdeanu et al., 2011; Burman et al., 2011).

Outside of Temporal KBP, there are several works on the task of extracting fluents from text. Wang et al. (2011) which uses label propagation, a graph-based semi-supervised method to extend positive and negative seed examples over the graph. Talukdar et al. (2012) apply a similar approach by aggregating local classification decisions using tempo-

[10] Percentages for "unknown" are omitted here.

ral constraints (e.g., mutual exclusion, containment, and succession) and joint inference. One key difference is that their constraints are included as input rather than learned by the system.

7 Conclusion and future Work

Joint inference can be effectively applied to the task of inferring timelines about named entities. Rather than using hard coded heuristics, our model learns and applies consistency constraints which capture inter-entity and cross-entity rules. Simple inference techniques such as random-restart hillclimbing score well and run efficiently. Both of our models (CC and JCC) obtain a substantial error reductions over simpler heuristics-based consistency approaches.

The overall framework can easily be applied to other information extraction tasks. Rather than listing rules for consistency, these can be learned and enforced via joint inference. While simple joint inference methods such as random-restart hillclimbing and Gibbs sampling worked well in our case, more complex inference methods may be required with more elaborate constraints.

A prime direction for future work is combining our model with a probabilistic relation extraction system. This could be accomplished by using the marginal probabilities on the extracted relations and multiplying them with the probabilities from the classifier and consistency components. Inference would require an additional step which could add or drop candidate fluents. Furthermore, the consistency component can be extended with new question types to incorporate non-temporal constraints as well.

Acknowledgments

The authors would like to thank the Stanford NLP group (with special thanks to Gabor Angeli and Mihai Surdeanu), William Headden, Micha Elsner, Pontus Stenetorp, and our anonymous reviewers for their helpful comments and feedback.

We gratefully acknowledge the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

References

James F. Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

Javier Artiles, Qi Li, Taylor Cassidy, Suzanne Tamang, and Heng Ji. 2011. CUNY BLENDER TAC-KBP2011 Temporal Slot Filling System Description. In *Proceedings of Text Analysis Conference (TAC)*, November.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 610–619, Portland, Oregon, USA, June. Association for Computational Linguistics.

Amev Burman, Arun Jayapal, Sathish Kannan, Madhu Kavilikatta, Ayman Alhelbawy, Leon Derczynski, and Robert Gaizauskas. 2011. USFD at KBP 2011: Entity Linking, Slot Filling and Temporal Bounding. In *Proceedings of Text Analysis Conference (TAC)*, November.

Lorna Byrne and John Dunnion. 2011. UCD IIRG at TAC 2011. In *Proceedings of Text Analysis Conference (TAC)*, November.

Nathanael Chambers and Dan Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 698–706. Association for Computational Linguistics.

Angel X. Chang and Christopher D. Manning. 2012. SUTIME: A library for recognizing and normalizing time expressions. In *8th International Conference on Language Resources and Evaluation (LREC 2012)*, May.

Ming-Wei Chang, Dan Goldwasser, Dan Roth, and Vivek Srikumar. 2010. Discriminative learning over constrained latent representations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 429–437. Association for Computational Linguistics.

Mark Craven and Johan Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 77–86. Heidelberg, Germany.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Proceedings of the COLING Workshop*

on Cross-framework and Cross-domain Parser Evaluation.

Jenny R. Finkel, Teg Grenager, and Christopher D. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

Guillermo Garrido, Bernardo Cabaleiro, Anselmo Pe nas, Alvaro Rodrigo, and Damiano Spina. 2011. A distant supervised learning system for the TAC-KBP Slot Filling and Temporal Slot Filling Tasks. In *Proceedings of Text Analysis Conference (TAC)*, November.

Heng Ji, Ralph Grishman, and Hoa Trang Dang. 2011. Overview of the TAC 2011 Knowledge Base Population track. In *Proceedings of Text Analysis Conference (TAC)*, November.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 423–430. Association for Computational Linguistics.

Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1288–1298. Association for Computational Linguistics.

Heeyoung Lee, Yves Peirsman, Angel X. Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's Multi-Pass Sieve Coreference Resolution System at the CoNLL-2011 Shared Task. In *CoNLL 2011*, page 28.

Xiao Ling and Daniel S. Weld. 2010. Temporal information extraction. In *Proceedings of the Twenty Fifth National Conference on Artificial Intelligence*.

Inderjeet Mani, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. 2006. Machine learning of temporal relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 753–760. Association for Computational Linguistics.

André F. T. Martins, Noah A. Smith, and Eric P. Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 342–350. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011. Association for Computational Linguistics.

Sebastian Riedel and Andrew McCallum. 2011. Fast and robust joint models for biomedical event extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, July.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 1–8, Boston, Massachusetts, USA, May 6 - May 7. Association for Computational Linguistics.

Mihai Surdeanu, Sonal Gupta, John Bauer, David McClosky, Angel X. Chang, Valentin I. Spitkovsky, and Christopher D. Manning. 2011. Stanford's Distantly-Supervised Slot-Filling System. In *Proceedings of Text Analysis Conference (TAC)*, November.

Partha Pratim Talukdar, Derry Wijaya, and Tom Mitchell. 2012. Coupled temporal scoping of relational facts. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, pages 73–82. ACM.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 173–180. Association for Computational Linguistics.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2008. A global joint model for semantic role labeling. *Computational Linguistics*, 34(2):161–191.

Yafang Wang, Bing Yang, Lizhen Qu, Marc Spaniol, and Gerhard Weikum. 2011. Harvesting facts from textual web sources by constrained label propagation. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 837–846. ACM.

Katsumasa Yoshikawa, Sebastian Riedel, Yuji Matsumoto, and Masayuki Asahara. 2009. Jointly identifying temporal relations with Markov Logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 405–413. Association for Computational Linguistics.