Implicit Argument Prediction with Event Knowledge

Pengxiang Cheng
Department of Computer Science
The University of Texas at Austin
pxcheng@cs.utexas.edu

Katrin Erk
Department of Linguistics
The University of Texas at Austin
katrin.erk@mail.utexas.edu

Abstract

Implicit arguments are not syntactically connected to their predicates, and are therefore hard to extract. Previous work has used models with large numbers of features, evaluated on very small datasets. We propose to train models for implicit argument prediction on a simple cloze task, for which data can be generated automatically at scale. This allows us to use a neural model, which draws on narrative coherence and entity salience for predictions. We show that our model has superior performance on both synthetic and natural data. [1]

1 Introduction

When parts of an event description in a text are missing, this event cannot be easily extracted, and it cannot easily be found as the answer to a question. This is the case with *implicit arguments*, as in this example from the reading comprehension dataset of Hermann et al. (2015):

> **Text:** More than 2,600 people have been infected by Ebola in Liberia, Guinea, Sierra Leone and Nigeria since the *outbreak* began in December, according to the World Health Organization. Nearly 1,500 have *died*.
>
> **Question:** The X outbreak has killed nearly 1,500.

In this example, it is Ebola that broke out, and Ebola was also the cause of nearly 1,500 people dying, but the text does not state this explicitly. *Ebola* is an implicit argument of both *outbreak* and *die*, which is crucial to answering the question.

We are particularly interested in implicit arguments that, like *Ebola* in this case, do appear in the text, but not as syntactic arguments of their predicates. Event knowledge is key to determining implicit arguments. In our example, diseases are maybe the single most typical things to *break out*, and diseases also typically kill people.

The task of identifying implicit arguments was first addressed by Gerber and Chai (2010) and Ruppenhofer et al. (2010). However, the datasets for the task were very small, and to our knowledge there has been very little further development on the task since then.

In this paper, we address the data issue by training models for implicit argument prediction on a simple cloze task, similar to the narrative cloze task (Chambers and Jurafsky, 2008), for which data can be generated automatically at scale. This allows us to train a neural network to perform the task, building on two insights. First, event knowledge is crucial for implicit argument detection. Therefore we build on models for narrative event prediction (Granroth-Wilding and Clark, 2016; Pichotta and Mooney, 2016a), using them to judge how coherent the narrative would be when we fill in a particular entity as the missing (implicit) argument. Second, the omitted arguments tend to be salient, as *Ebola* is in the text from which the above example is taken. So in addition to narrative coherence, our model takes into account entity salience (Dunietz and Gillick, 2014).

In an evaluation on a large automatically generated dataset, our model clearly outperforms even strong baselines, and we find salience features to be important to the success of the model. We also evaluate against a variant of the Gerber and Chai (2012) model that does not rely on gold features, finding that our simple neural model outperforms their much more complex model.

Our paper thus makes two major contributions. 1) We propose an argument cloze task to generate synthetic training data at scale for implicit argument prediction. 2) We show that neural event

[1] Our code is available at https://github.com/pxch/event_imp_arg.

Proceedings of NAACL-HLT 2018, pages 831–840
New Orleans, Louisiana, June 1 - 6, 2018. ©2018 Association for Computational Linguistics

models for narrative schema prediction can be used on implicit argument prediction, and that a straightforward combination of event knowledge and entity salience can do well on the task.

2 Related Work

While dependency parsing and semantic role labeling only deal with arguments that are available in the syntactic context of the predicate, implicit argument labeling seeks to find argument that are not syntactically connected to their predicates, like *Ebola* in our introductory example.

The most relevant work on implicit argument prediction came from Gerber and Chai (2010), who built an implicit arguments dataset by selecting 10 nominal predicates from NomBank (Meyers et al., 2004) and manually annotating implicit arguments for all occurrences of these predicates. In an analysis of their data they found implicit arguments to be very frequent, as their annotation added 65% more arguments to NomBank. Gerber and Chai (2012) also trained a linear classifier for the task relying on many hand-crafted features, including gold features from FrameNet (Baker et al., 1998), PropBank (Palmer et al., 2005) and NomBank. This classifier has, to the best of our knowledge, not been outperformed by follow-up work (Laparra and Rigau, 2013; Schenk and Chiarcos, 2016; Do et al., 2017). We evaluate on the Gerber and Chai dataset below. Ruppenhofer et al. (2010) also introduced an implicit argument dataset, but we do not evaluate on it as it is even smaller and much more complex than Gerber and Chai (2010). More recently, Modi et al. (2017) introduced the referent cloze task, in which they predicted a manually removed discourse referent from a human annotated narrative text. This task is closely related to our argument cloze task.

Since we intend to exploit event knowledge in predicting implicit arguments, we here refer to recent work on statistical script learning, started by Chambers and Jurafsky (2008, 2009). They introduced the idea of using statistical information on coreference chains to induce prototypical sequences of narrative events and participants, which is related to the classical notion of a script (Schank and Abelson, 1977). They also proposed the narrative cloze evaluation, in which one event is removed at random from a sequence of narrative events, then the missing event is predicted given all context events. We use a similar trick to define a cloze task for implicit argument prediction, discussed in Section 3.

Many follow-up papers on script learning have used neural networks. Rudinger et al. (2015) showed that sequences of events can be efficiently modeled by a log-bilinear language model. Pichotta and Mooney (2016a,b) used an LSTM to model a sequence of events. Granroth-Wilding and Clark (2016) built a network that produces an event representation by composing its components. To do the cloze task, they select the most probable event based on pairwise event coherence scores. For our task we want to do something similar: We want to predict how coherent a narrative would be with a particular entity candidate filling the implicit argument position. So we take the model of Granroth-Wilding and Clark (2016) as our starting point.

The Hermann et al. (2015) reading comprehension task, like our cloze task, requires systems to guess a removed entity. However in their case the entity is removed in a summary, not in the main text. In their case, the task typically amounts to finding a main text passage that paraphrases the sentence with the removed entity; this is not the case in our cloze task.

3 The Argument Cloze Task

We present the **argument cloze** task, which allows us to automatically generate large scale data for training (Section 6.1) and evaluation (Section 5.1).

In this task, we randomly remove an entity from an argument position of one event in the text. The entity in question needs to appear in at least one other place in the text. The task is then for the model to pick, from all entities appearing in the text, the one that has been removed. We first define what we mean by an event, then what we mean by an entity. Like Pichotta and Mooney (2016a); Granroth-Wilding and Clark (2016), we define an *event* e as consisting of a verbal predicate v, a subject s, a direct object o, and a prepositional object p (along with the preposition). Here we only allow one prepositional argument in the structure, to avoid variable length input in the event composition model.[2] By an *entity*, we mean a coreference chain with a length of at least two – that is, the entity needs to appear at least twice in the text.

For example, from a piece of raw text (Figure

[2]In case of multiple prepositional objects, we select the one that is closest to the predicate.

Manville Corp. said it will build a $ 24 million power plant to provide electricity to its Igaras pulp and paper mill in Brazil .

The company said the plant will ensure that it has adequate energy for the mill and will reduce the mill's energy costs .

(a) A piece of raw text from OntoNotes corpus.

x_0 = The company x_1 = mill x_2 = power plant

e_0: (*build-pred*, x_0-*subj*, x_2-*dobj*, —)
e_1: (*provide-pred*, —, *electricity-dobj*, x_1-*prep_to*)
e_2: (*ensure-pred*, x_2-*subj*, —, —)
e_3: (*has-pred*, x_0-*subj*, *energy-dobj*, x_1-*prep_for*)
e_4: (*reduce-pred*, x_2-*subj*, *cost-dobj*, —)

(b) Extracted events (e_0~e_4) and entities (x_0~x_2), using gold annotations from OntoNotes.

e_0, e_2, e_3, e_4: same as above
e_1: (*provide-pred*, —, *electricity-dobj*, ***??-prep_to***)

x_0 = The company x_1 = mill x_2 = power plant

(c) Example of an argument cloze task for *prep_to* of e_1.

Figure 1: Example of automatically extracted events and entities and an argument cloze task.

1a), we automatically extract a sequence of events from a dependency parse, and a list of entities from coreference chains. In Figure 1b, e_0~e_4 are events, x_0~x_2 are entities. The arguments *electricity-dobj* and *energy-dobj* are not in coreference chains and are thus not candidates for removal. An example of the argument cloze task is shown in Figure 1c. Here the *prep_to* argument of e_1 has been removed.

Coreference resolution is very noisy. Therefore we use gold coreference annotation for creating evaluation data, but automatically generated coreference chains for creating training data.

4 Methods

4.1 Modeling Narrative Coherence

We model implicit argument prediction as selecting the entity that, when filled in as the implicit argument, makes the overall most coherent narrative. Suppose we are trying to predict the direct object argument of some target event e_t. Then we complete e_t by putting an entity candidate into the direct object argument position, and check the coherence of the resulting event with the rest of the narrative. Say we have a sequence of events $e_1, e_2, \ldots, e_n$ in a narrative, and a list of entity candidates $x_1, x_2, \ldots, x_m$. Then for any candidate x_j, we first complete the target event to be

$$e_t(j) = (v_t, s_t, x_j, p_t), \quad j = 1, \ldots, m \qquad (1)$$

where v_t, s_t, and p_t are the predicate, subject, and prepositional object of e_t respectively, and x_j is filled as the direct object. (Event completion for omitted subjects and prepositional objects is analogous.)

Then we compute the narrative coherence score S_j of the candidate x_j by[3]

$$S_j = \max_{c=1,\, c\neq t}^{n} coh\left(\vec{e_t(j)}, \vec{e_c}\right), \quad j = 1, \ldots, m \qquad (2)$$

where $\vec{e_t(j)}$ and $\vec{e_c}$ are representations for the completed target event $e_t(j)$ and one context event e_c, and coh is a function computing a coherence score between two events, both depending on the model being used. The candidate x_j with the highest score S_j is then selected as our prediction.

4.2 The Event Composition Model

To model coherence (coh) between a context event and a target event, we build an event composition model consisting of three parts, as shown in Figure 2: event components are representated through **event-based word embeddings**, which encode event knowledge in word representations; the **argument composition network** combines the components to produce event representations; and the **pair composition network** compute a coherence score for two event representations.

This basic architecture is as in the model of Granroth-Wilding and Clark (2016). However our model is designed for a different task, argument cloze rather than narrative cloze, and for our task entity-specific information is more important. We therefore create the training data in a different way, as described in Section 4.2.1. We now discuss the three parts of the model in more detail.

Event-Based Word Embeddings The model takes word embeddings of both predicates and

[3] We have also tried using the sum instead of the maximum, but it did not perform as well across different models and datasets.

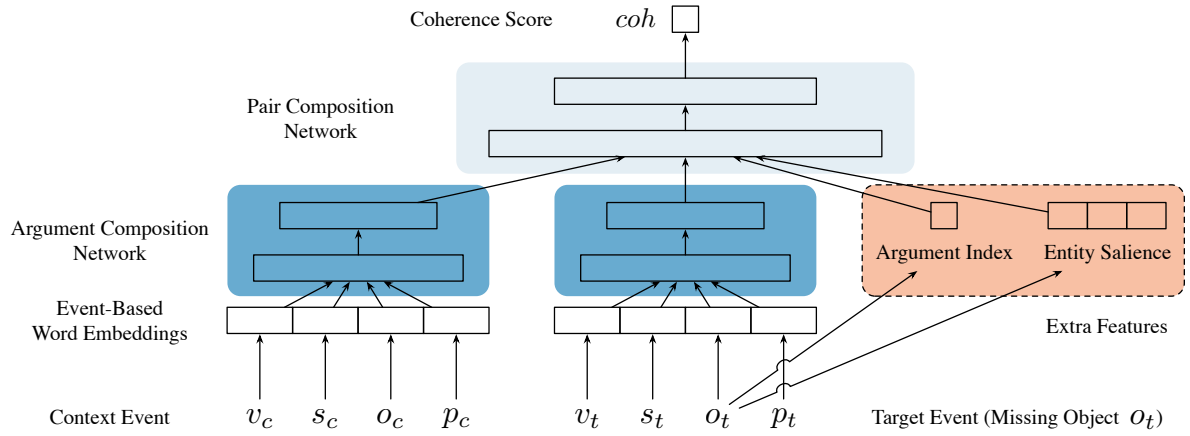


Figure 2: Diagram for event composition model. ***Input***: a context event and a target event. ***Event-Based Word Embeddings***: embeddings for components of both events that encodes event knowledge. ***Argument Composition Network***: produces an event representation from its components. ***Pair Composition Network***: computes a coherence score *coh* from two event representations. ***Extra Features***: argument index and entity salience features as additional input to the pair composition network.

arguments as input to compute event representations. To better encode event knowledge in word level, we train an SGNS (skip-gram with negative sampling) word2vec model (Mikolov et al., 2013) with event-specific information. For each extracted event sequence, we create a sentence with the predicates and arguments of all events in the sequence. An example of such a training sentence is given in Figure 3.

build-pred company-subj plant-dobj provide-pred electricity-dobj mill-prep_to ensure-pred plant-*subj has-pred company-subj energy-dobj mill-prep_for reduce-pred plant-subj cost-dobj*

Figure 3: Event-based word2vec training sentence, constructed from events and entities in Figure 1b.

Argument Composition Network The argument composition network (dark blue area in Figure 2) is a two-layer feedforward neural network that composes an event representation from the embeddings of its components. Non-existent argument positions are filled with zeros.

Pair Composition Network The pair composition network (light blue area in Figure 2) computes a coherence score *coh* between 0 and 1, given the vector representations of a context event and a target event. The coherence score should be high when the target event contains the correct argument, and low otherwise. So we construct the training objective function to distinguish the correct argument from wrong ones, as described in Equation 3.

4.2.1 Training for Argument Prediction

To train the model to pick the correct candidate, we automatically construct training samples as event triples consisting of a context event e_c, a positive event e_p, and a negative event e_n. The context event and positive event are randomly sampled from an observed sequence of events, while the negative event is generated by replacing one argument of positive event by a random entity in the narrative, as shown in Figure 4.

x_0 = The company x_1 = mill x_2 = power plant

Context: (*build-pred*, x_0-*subj*, x_2-*dobj*, –)
Positive: (*reduce-pred*, **x_2-*subj***, *cost-dobj*, –)
Negative: (*reduce-pred*, **x_1-*subj***, *cost-dobj*, –)

Figure 4: Example of an event triple constructed from events and entities in Figure 1b.

We want the coherence score between e_c and e_p to be close to 1, while the score for e_c and e_n should be close to 0. Therefore, we train the model to minimize cross-entropy as follows:

$$\frac{1}{m}\sum_{i=1}^{m} -\log(coh(e_{ci}, e_{pi})) - \log(1 - coh(e_{ci}, e_{ni})) \quad (3)$$

where e_{ci}, e_{pi}, and e_{ni} are the context, positive, and negative events of the ith training sample respectively.

4.3 Entity Salience

Implicit arguments tend to be salient entities in the document. So we extend our model by entity salience features, building on recent work by Dunietz and Gillick (2014), who introduced a simple model with several surface level features for entity salience detection. Among the features they used, we discard those that require external resources, and only use the remaining three features, as illustrated in Table 1. Dunietz and Gillick found *mentions* to be the most powerful indicator for entity salience among all features. We expect similar results in our experiments, however we include all three features in our event composition model for now, and conduct an ablation test afterwards.

Feature	Description
1st_loc	Index of the sentence where the first mention of the entity appears
head_count	Number of times the head word of the entity appears
mentions	A vector containing the numbers of named, nominal, pronominal, and total mentions of the entity

Table 1: Entity salience features from Dunietz and Gillick (2014).

The entity salience features are directly passed into the pair composition network as additional input. We also add an extra feature for argument position index (encoding whether the missing argument is a subject, direct object, or prepositional object), as shown in the red area in Figure 2.

5 Evaluation Datasets

5.1 Argument Cloze Evaluation

Previous implicit argument datasets were very small. To overcome that limitation, we automatically create a large and comprehensive evaluation dataset, following the **argument cloze** task setting in Section 3.

Since the events and entities are extracted from dependency labels and coreference chains, we do not want to introduce systematic error into the evaluation from imperfect parsing and coreference algorithms. Therefore, we create the evaluation set from OntoNotes (Hovy et al., 2006), which contains human-labeled dependency and coreference annotation for a large corpus. So the extracted events and entities in the evaluation set are gold. Note that this is only for evaluation; in training we do not rely on any gold annotations (Section 6.1).

There are four English sub-corpora in OntoNotes Release 5.0[4] that are annotated with dependency labels and coreference chains. Three of them, which are mainly from broadcast news, share similar statistics in document length, so we combine them into a single dataset and name it **ON-SHORT** as it consists mostly of short documents. The fourth subcorpus is from the *Wall Street Journal* and has significantly longer documents. We call this subcorpus **ON-LONG** and evaluate on it separately. Some statistics are shown in Table 2.

	ON-SHORT	ON-LONG
# doc	1027	597
# test cases	13018	18208
Avg # entities	12.06	36.95

Table 2: Statistics on argument cloze datasets.

5.2 The Gerber and Chai (G&C) Dataset

The implicit argument dataset from Gerber and Chai (2010) (referred as **G&C** henceforth) consists of 966 human-annotated implicit argument instances on 10 nominal predicates.

To evaluate our model on G&C, we convert the annotations to the input format of our model as follows: We map nominal predicates to their verbal form, and semantic role labels to syntactic argument types based on the NomBank frame definitions. One of the examples (after mapping semantic role labels) is as follows:

> [Participants]$_{subj}$ will be able to transfer [money]$_{dobj}$ to [other investment funds]$_{prep\_to}$. The [investment]$_{pred}$ choices are limited to [a stock fund and a money-market fund]$_{prep\_to}$.

For the nominal predicate *investment*, there are three arguments missing (*subj*, *dobj*, *prep_to*). The model first needs to determine that each of those argument positions in fact has an implicit filler. Then, from a list of candidates (not shown here), it

[4]LDC Catalog No. LDC2013T19

needs to select *Participants* as the implicit *subj* argument, *money* as the implicit *dobj* argument, and either *other investment funds* or *a stock fund and a money-market fund* as the implicit *prep_to*.

6 Experiments

6.1 Implementation Details

We train our neural model using synthetic data as described in Section 3. For creating the training data, we do not use gold parses or gold coreference chains. We use the 20160901 dump of English Wikipedia[5], with 5,228,621 documents in total. For each document, we extract plain text and break it into paragraphs, while discarding all structured data like lists and tables[6]. We construct a sequence of events and entities from each paragraph, by running Stanford CoreNLP (Manning et al., 2014) to obtain dependency parses and coreference chains. We lemmatize all verbs and arguments. We incorporate negation and particles in verbs, and normalize passive constructions. We represent each argument by the corresponding entity's representative mention if it is linked to an entity, otherwise by its head lemma. We keep verbs and arguments with counts over 500, together with the 50 most frequent prepositions, leading to a vocabulary of 53,345 tokens; all other words are replaced with an out-of-vocabulary token. The most frequent verbs (with counts over 100,000) are down-sampled.

For training the event-based word embeddings, we create pseudo-sentences (Section 4.2) from all events of all sequences (approximately 87 million events) as training samples. We train an SGNS word2vec model with embedding size = 300, window size = 10, subsampling threshold = 10^{-4}, and negative samples = 10, using the Gensim package (Řehůřek and Sojka, 2010).

For training the event composition model, we follow the procedure described in Section 4.2.1, and extract approximately 40 million event triples as training samples[7]. We use a two-layer feed-forward neural network with layer sizes 600 and 300 for the argument composition network, and another two-layer network with layer sizes 400 and 200 for the pair composition network. We use cross-entropy loss with ℓ_2 regularization of 0.01. We train the model using stochastic gradient descent (SGD) with a learning rate of 0.01 and a batch size of 100 for 20 epochs.

To study how the size of the training set affects performance, we downsample the 40 million training samples to another set of 8 million training samples. We refer to the resulting models as **EVENTCOMP-8M** and **EVENTCOMP-40M**.

6.2 Evaluation on Argument Cloze

For the synthetic argument cloze task, we compare our model with 3 baselines.

RANDOM Randomly select one entity from the candidate list.

MOSTFREQ Always select the entity with highest number of mentions.

EVENTWORD2VEC Use the event-based word embeddings described in Section 4.2 for predicates and arguments. The representation of an event e is the sum of the embeddings of its components, i.e.,

$$\vec{e} = \vec{v} + \vec{s} + \vec{o} + \vec{p} \tag{4}$$

where $\vec{v}, \vec{s}, \vec{o}, \vec{p}$ are the embeddings of verb, subject, object, and prepositional object, respectively. The coherence score of two events in this baseline model is their cosine similarity. Like in our main model, the coherence score of the candidate is then the maximum pairwise coherence score, as described in Section 4.1.

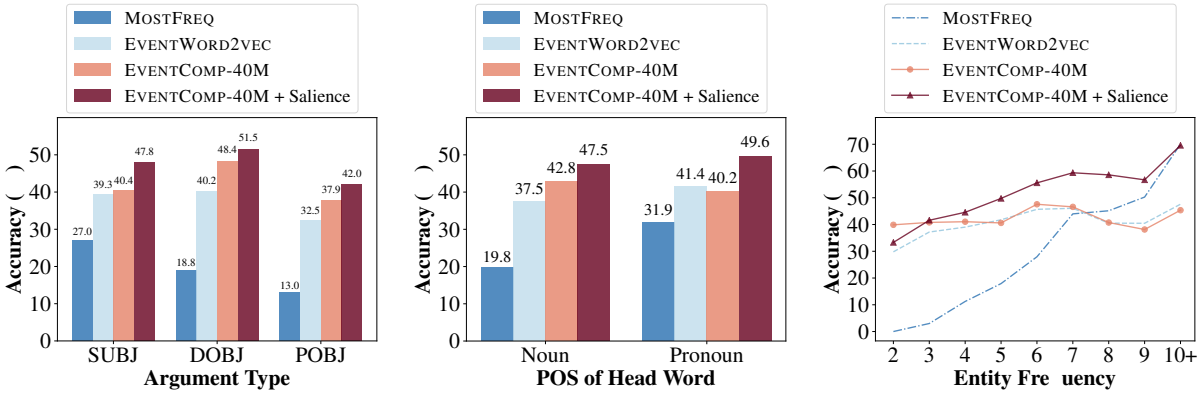
The evaluation results on the ON-SHORT dataset are shown in Table 3. The EVENTWORD2VEC baseline is much stronger than the other two, achieving an accuracy of 38.40%. In fact, EVENTCOMP-8M by itself does not do better than EVENTWORD2VEC, but adding entity salience greatly boosts performance. Using more training data (EVENTCOMP-40M) helps by a substantial margin both with and without entity salience features.

To see which of the entity salience features are important, we conduct an ablation test with the EVENTCOMP-8M model on ON-SHORT. From the results in Table 4, we can see that in our task, as in Dunietz and Gillick (2014), the entity mentions features, i.e., the numbers of named, nominal, pronominal, and total mentions of the entity, are most helpful. In fact, the other two features even decrease performance slightly.

[5] https://dumps.wikimedia.org/enwiki/

[6] We use the WikiExtractor tool at https://github.com/attardi/wikiextractor.

[7] We only sample one negative event for each pair of context and positive events for fast training, though more training samples are easily accessible.



(a) Accuracy by Argument Type (b) Accuracy by POS of Head Word (c) Accuracy by Entity Frequency

Figure 5: Performance of EventComp (with and without entity salience) and two baseline models by (a) argument type, (b) part-of-speech tag of the head word of the entity, and (c) entity frequency.

	Accuracy (%)
Random	8.29
MostFreq	22.76
EventWord2vec	38.40
EventComp-8M	38.26
+ entity salience	45.05
EventComp-40M	41.89
+ entity salience	**47.75**

Table 3: Evaluation on ON-Short.

Features	Accuracy (%)
no entity salience feature	38.26
– *mentions*	39.02
– *head_count*	**45.71**
– *1st_loc*	**45.65**
all entity salience features	45.05

Table 4: Ablation test on entity salience features. (Using EventComp-8M on ON-Short.)

We take a closer look at several of the models in Figure 5. Figure 5a breaks down the results by the argument type of the removed argument. On subjects, the EventWord2vec baseline matches the performance of EventComp, but not on direct objects and prepositional objects. Subjects are semantically much less diverse than the other argument types, as they are very often animate. A similar pattern is apparent in Figure 5b, which has results by the part-of-speech tag of the head word of the removed entity. Note that an entity is a coreference chain, not a single mention; so when the head word is a pronoun, this is an entity which has only pronoun mentions. A pronoun entity provides little semantic content beyond, again, animacy. And again, EventWord2vec performs well on pronoun entities, but less so on entities described by a noun. It seems that EventWord2vec can pick up on a coarse-grained pattern such as animate/inanimate, but not on more fine-grained distinctions needed to select the right noun, or to select a fitting direct object or prepositional object. This matches the fact that EventWord2vec gets a less clear signal on the task, in two respects: It gets much less information than EventComp on the distinction between argument positions,[8] and it only looks at overall event similarity while EventComp is trained to detect narrative coherence. Entity salience contributes greatly across all argument types and parts of speech, but more strongly on subjects and pronouns. This is again because subjects, and pronouns, are semantically less distinct, so they can only be distinguished by relative salience.

Figure 5c analyzes results by the frequency of the removed entity, that is, by its number of mentions. The MostFreq baseline, unsurprisingly, only does well when the removed entity is a highly frequent one. The EventComp model is much better than MostFreq at picking out the right entity when it is a rare one, as it can look at the semantic content of the entity as well as its frequency. Entity salience boosts the performance of EventComp in particular for frequent entities.

The ON-Long dataset, as discussed in Section 5.1, consists of OntoNotes data with much

[8] As shown in Figure 3, the "words" for which embeddings are computed are role-lemma pairs.

longer documents than found in ON-SHORT. Evaluation results on ON-LONG are shown in Table 5. Although the overall numbers are lower than those for ON-SHORT, we are selecting from 36.95 candidates on average, more than 3 times more than for ON-SHORT. Considering that the accuracy of randomly selecting an entity is as low as 2.71%, the performance of our best performing model, with an accuracy of 27.87%, is quite good.

	Accuracy (%)
RANDOM	2.71
MOSTFREQ	17.23
EVENTWORD2VEC	21.49
EVENTCOMP-8M	18.79
+ entity salience	26.23
EVENTCOMP-40M	21.79
+ entity salience	**27.87**

Table 5: Evaluation on ON-LONG.

6.3 Evaluation on G&C

The G&C data differs from the Argument Cloze data in two respects. First, not every argument position that seems to be open needs to be filled: The model must additionally make a **fill / no-fill decision**. Whether a particular argument position is typically filled is highly predicate-specific. As the small G&C dataset does not provide enough data to train our neural model on this task, we instead train a simple logistic classifier, the **fill / no-fill classifier**, with a small subset of shallow lexical features used in Gerber and Chai (2012), to make the decision. These features describe the syntactic context of the predicate. We use only 14 features; the original Gerber and Chai model had more than 80 features, and our re-implementation, described below, has around 60.

The second difference is that in G&C, an event may have multiple open argument positions. In that case, the task is not just to select a candidate entity, but also to determine which of the open argument positions it should fill. So the model must do **multi implicit argument prediction**. We can flexibly adapt our method for training data generation to this case. In particular, we create extra negative training events, in which an argument of the positive event has been moved to another argument position in the same event, as shown in Figure 6. We can then simply train our EVENTCOMP model on this extended training data. We refer to the extra training process as **multi-arg training**.

x_0 = The company x_1 = mill x_2 = power plant

Context: (*build-pred*, x_0-*subj*, x_2-*dobj*, —)
Positive: (*reduce-pred*, **x_2-*subj***, *cost-dobj*, —)
Negative: (*reduce-pred*, —, *cost-dobj*, **x_2-*prep***)

Figure 6: Event triples for training multi implicit argument prediction.

We compare our models to that of Gerber and Chai (2012). However, their original logistic regression model used many features based on gold annotation from FrameNet, PropBank and NomBank. To create a more realistic evaluation setup, we re-implement a variant of their original model by removing gold features, and name it GCAUTO. Results from GCAUTO are directly comparable to our models, as both are trained on automatically generated features. [9]

	P	R	F_1
Gerber and Chai (2012)	57.9	44.5	50.3
GCAUTO	49.9	40.1	44.5
EVENTCOMP-8M	8.9	27.9	13.5
+ fill / no-fill classifier	22.0	22.3	22.1
+ multi-arg training	43.5	44.1	43.8
+ entity salience	45.7	46.4	**46.1**
EVENTCOMP-40M	9.4	30.3	14.3
+ fill / no-fill classifier	23.7	24.0	23.9
+ multi-arg training	46.7	47.3	47.0
+ entity salience	49.3	49.9	**49.6**

Table 6: Evaluation on G&C dataset.

We present the evaluation results in Table 6. The original EVENTCOMP models do not perform well, which is as expected since the model is not designed to do the *fill / no-fill decision* and *multi implicit argument prediction* tasks as described above. With the fill / no-fill classifier, precision rises by around 13 points because this classifier prevents many false positives. With additional multi-arg training, F_1 score improves by another 22-23 points. At this point, our model

[9] To be fair, we also tested adding the fill / no-fill classifier to GCAUTO. However the classifier only increases precision at the cost of reducing recall, and GCAUTO already has higher precision than recall. The resulting F_1 score is actually worse, and thus is not reported here.

achieves a performance comparable to the much more complex G&C reimplementation GCAUTO. Adding entity salience features further boosts both precision and recall, showing that implicit arguments do tend to be filled by salient entities, as we had hypothesized. Again, more training data substantially benefits the task. Our best performing model, at 49.6 F_1, clearly outperforms GCAUTO, and is comparable with the original Gerber and Chai (2012) model trained with gold features. [10]

7 Conclusion

In this paper we have addressed the task of implicit argument prediction. To support training at scale, we have introduced a simple cloze task for which data can be generated automatically. We have introduced a neural model, which frames implicit argument prediction as the task of selecting the textual entity that completes the event in a maximally narratively coherent way. The model prefers salient entities, where salience is mainly defined through the number of mentions. Evaluating on synthetic data from OntoNotes, we find that our model clearly outperforms even strong baselines, that salience is important throughout for performance, and that event knowledge is particularly useful for the (more verb-specific) object and prepositional object arguments. Evaluating on the naturally occurring data from Gerber and Chai, we find that in a comparison without gold features, our model clearly outperforms the previous state-of-the-art model, where again salience information is important.

The current paper takes a first step towards predicting implicit arguments based on narrative coherence. We currently use a relatively simple model for local narrative coherence; in the future we will turn to models that can test global coherence for an implicit argument candidate. We also plan to investigate how the extracted implicit arguments can be integrated into a downstream task that makes use of event information, in particular we would like to experiment with reading comprehension.

Acknowledgments

This research was supported by NSF grant IIS 1523637. We also acknowledge the Texas Advanced Computing Center for providing grid resources that contributed to these results, and we would like to thank the anonymous reviewers for their valuable feedback.

[10] We also tried fine tune our model on the G&C dataset with cross validation, but the model severely overfit, possibly due to the very small size of the dataset.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*. http://www.aclweb.org/anthology/P98-1013.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*. Association for Computational Linguistics, pages 789–797. http://www.aclweb.org/anthology/P08-1090.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. 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*. Association for Computational Linguistics, pages 602–610. http://www.aclweb.org/anthology/P09-1068.

Quynh Ngoc Thi Do, Steven Bethard, and Marie-Francine Moens. 2017. Improving implicit semantic role labeling by predicting semantic frame arguments. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Asian Federation of Natural Language Processing, pages 90–99. http://www.aclweb.org/anthology/I17-1010.

Jesse Dunietz and Daniel Gillick. 2014. A new entity salience task with millions of training examples. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*. Association for Computational Linguistics, pages 205–209. https://doi.org/10.3115/v1/E14-4040.

Matthew Gerber and Joyce Chai. 2010. Beyond NomBank: A study of implicit arguments for nominal predicates. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 1583–1592. http://www.aclweb.org/anthology/P10-1160.

Matthew Gerber and Joyce Y. Chai. 2012. Semantic role labeling of implicit arguments for nominal predicates. *Computational Linguistics* 38(4). https://doi.org/10.1162/COLI_a_00110.

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. In *AAAI Conference on Artificial Intelligence*. pages 2727–2733.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. pages 1693–1701.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. http://www.aclweb.org/anthology/N06-2015.

Egoitz Laparra and German Rigau. 2013. ImpAr: A deterministic algorithm for implicit semantic role labelling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 1180–1189. http://www.aclweb.org/anthology/P13-1116.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Association for Computational Linguistics, pages 55–60. https://doi.org/10.3115/v1/P14-5010.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*. pages 24–31. http://www.aclweb.org/anthology/W04-2705.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.

Ashutosh Modi, Ivan Titov, Vera Demberg, Asad Sayeed, and Manfred Pinkal. 2017. Modelling semantic expectation: Using script knowledge for referent prediction. *Transactions of the Association of Computational Linguistics* 5:31–44. http://www.aclweb.org/anthology/Q17-1003.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics* 31(1). http://www.aclweb.org/anthology/J05-1004.

Karl Pichotta and Raymond J Mooney. 2016a. Learning statistical scripts with LSTM recurrent neural networks. In *AAAI Conference on Artificial Intelligence*. pages 2800–2806.

Karl Pichotta and Raymond J. Mooney. 2016b. Using sentence-level LSTM language models for script inference. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 279–289. https://doi.org/10.18653/v1/P16-1027.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, pages 45–50. http://is.muni.cz/publication/884893/en.

Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script induction as language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1681–1686. https://doi.org/10.18653/v1/D15-1195.

Josef Ruppenhofer, Caroline Sporleder, Roser Morante, Collin Baker, and Martha Palmer. 2010. SemEval-2010 Task 10: Linking events and their participants in discourse. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, pages 45–50. http://www.aclweb.org/anthology/S10-1008.

Roger C Schank and Robert Abelson. 1977. Scripts, goals, plans, and understanding.

Niko Schenk and Christian Chiarcos. 2016. Unsupervised learning of prototypical fillers for implicit semantic role labeling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 1473–1479. https://doi.org/10.18653/v1/N16-1173.