Simple Large-scale Relation Extraction from Unstructured Text

Christos Christodoulopoulos, Arpit Mittal
Amazon Research Cambridge
{chrchrs, mitarpit}@amazon.co.uk

Abstract

Knowledge-based question answering relies on the availability of facts, the majority of which cannot be found in structured sources (e.g. Wikipedia info-boxes, Wikidata). One of the major components of extracting facts from unstructured text is Relation Extraction (RE). In this paper we propose a novel method for creating distant (weak) supervision labels for training a large-scale RE system. We also provide new evidence about the effectiveness of neural network approaches by decoupling the model architecture from the feature design of a state-of-the-art neural network system. Surprisingly, a much simpler classifier trained on similar features performs on par with the highly complex neural network system (at 75x reduction to the training time), suggesting that the features are a bigger contributor to the final performance.

Keywords: relation extraction, distant supervision, unstructured text

1. Introduction

Knowledge-based question answering relies on the availability of facts – usually in the form of triples, stored in large-scale knowledge bases (KBs) e.g. Freebase (Bollacker et al., 2008), DBPedia (Auer et al., 2007). There are two main sources of facts for such a KB: structured data (e.g. Wikipedia info-boxes, Wikidata) or unstructured text. Undeniably, the former type of knowledge extraction is very accurate and has been the main source of knowledge behind the major industrial knowledge bases. However, the facts extracted from structured sources cover a limited set of high-importance relations, leaving a large number of them implicitly (or explicitly) mentioned in unstructured text (McCallum, 2005).

In order to ground the following presentation, we will present a typical problem from the factual knowledge extraction domain with the following unstructured text from a Wikipedia page:

> "*Carrie Fisher wrote several semi-autobiographical novels, including Postcards from the Edge.*"

The purpose of a fact extraction system is to extract the following facts of the form of `predicate` (*subject*, *object*): **instance of (postcards from the edge, novel)**, and **author of (postcards from the edge, carrie fisher)**, where the first part is a *relation*, and the other parts are the left and right *entities* participating in that relation.

Typically three tasks are involved in generating facts: Entity Recognition, Entity Resolution (or Entity Linking), and Relation Extraction (RE). Entity Recognition and Resolution deal with the task of translating surface strings to KB entities. This includes nominal or pronominal coreference resolution: we should be able to extract the same entity even if the text stated that 'Fisher wrote…' (instead of resolving e.g. to Bobby Fisher) or 'She wrote…' (provided that Carrie Fisher's name was mentioned in a previous sentence). Relation Extraction extracts relation triples (or facts) involving those entities with appropriate relations (also part of the KB schema). Each of these components could be built and operated in isolation, but they affect the performance of each other.

In this paper, we examine the task of RE focusing on extracting knowledge to enrich a large-scale KB (∼billions of facts). We consider a state-of-the-art model that has been applied to hyponymy detection and present a thorough analysis of its application to datasets derived from Wikidata and Alexa KB, a proprietary large-scale triple KB that powers Amazon's Alexa. We also present a new way of generating distant supervision for relation extraction with a simple yet effective way of reducing the noise for the entity resolution.

2. Related work

Relation Extraction is the NLP task of extracting structured semantic relations between entities from natural (unstructured) text. Formally, it can be defined as identifying semantic relations between (resolved) entities and normalise these relations by mapping them to a predefined KB schema. In the NLP community, the RE task evolved out of the Information Extraction projects like MUC in the 1990s (see Chinchor et al. (1993) for an overview) and ACE in the 2000s (Doddington et al., 2004). In both projects the main focus was the automatic extraction of *events* rather than relations (the main difference being that an event is a special type of fact that involves actor entities and occurs at a specific time point) in a limited set of domains (e.g. bombings, company mergers, etc.). This meant that in both projects the number of relations marked for extraction was very limited (3 relations in MUC and 24 in ACE with 7k relation instances for 40k entity mentions).

Starting with those projects, the task of RE was thought of as a pipeline, where the entities were first detected, resolved to a standard schema, and then the RE system would determine which of the possible relations was expressed (if any) between any given pair of entities. Much of the earlier work explored a variety of different features, such as syntactic phrase chunking and constituency parsing (Bunescu and Mooney, 2005; Jiang and Zhai, 2007; Qian et al., 2008), and semantic knowledge like WordNet (Zhou et al., 2005), although Jiang and Zhai (2007) showed that the more complex features might actually hurt the performance

of an SVM-based RE system. The work of Shwartz et al. (2016), that we closely follow, is also using both semantic and syntactic features, by combining the dependency paths between entities, with word embedding representations of both the entities and the lemmas in the dependency paths.

Another related area is relation extraction for Open Information Extraction (OpenIE). Some of the more representative projects in the area, like Reverb (Fader et al., 2011) and more recently ClauseIE (Del Corro and Gemulla, 2013) use syntactic information (PoS tagging / chunking, and dependency parsing respectively) to extract entity and relation phrases. However, unlike OpenIE, we are interested in normalized entities and relations (i.e. that map to a knowledge base).

In this work, we follow a common way of producing training examples for RE is to use *distant supervision* (Craven et al., 1999; Mintz et al., 2009): the assumption is that if any sentence mentions two entities which we know (from a KB) participate in a specific relation, that sentence must be evidence for that relation. In the area of distant supervision, there are two relevant research directions. The first is to use it for directly enriching KBs from unstructured text, as well as leverage the KBs to generate the distant supervision labels (Poon et al., 2015; Parikh et al., 2015). The second direction attempts to reduce the noise in distant supervision labels. A first line of approaches, starting with Data Programming (Ratner et al., 2016), uses generative models to combine multiple sources of weak supervision (e.g. automatically extracted from a KB, rules generated by experts etc.) in order to predict disagreements and overlaps between them and create a noise-aware posterior distribution of predictions. An extension of this approach is Socratic Learning (Varma et al., 2017) which uses the differences in the predictions of the generative model and the main classification system to discover discriminating features and add them back to the generative model. As these approaches require multiple sources of weak supervision, we examine another line of projects which works by aggregating the support sentences[1] for each entity pair (Riedel et al., 2010; Hoffmann et al., 2011). This is the approach that Shwartz et al. (2016) and the current work follow.

2.1. HypeNET

A recent paper (Shwartz et al., 2016) proposed HypeNET, a new method for RE that integrated dependency path information with distributional semantic vector representation of the entities. The authors applied this method to extract hyponyms (i.e. **instance of** relations) and also made a new version of their system publicly available.[2] The training examples used (entity/relation triples) come from a number of sources like WordNet (Miller, 1995), Yago (Hoffart et al., 2013), DBPedia and Wikidata, and the source of the linguistic features (part-of-speech tags, dependency paths, noun phrases) was the 2015 dump of Wikipedia, processed using the spaCy system[3]. Their proposed system achieved by far the best results on their dataset. Since **instance of** is one of the most often used relations (most of the uses are implicit, during inference), we decided to investigate HypeNET as the base of our RE system.

The training examples used by the authors of HypeNET consisted of facts about only one relation. We wanted to build a system that works on multiple relations at a very large scale. Hence, in this work we use two different dataset sources: Wikidata, a publicly-available large-scale KB to aid reproducibility, as well as the larger Alexa KB, built by combining a hand-curated ontology with publicly available data from Wikidata, Wikipedia, Freebase, DBPedia, and other sources.

3. Distant supervision

Following the technique presented in Mintz et al. (2009), and the implementation in HypeNET, we needed to generate training examples where entities X and Y are connected by a relation in the KB and also appear together in the same sentence. When we applied the distant supervision technique presented in HypeNET to our datasets (both Wikidata and Alexa KB) we got poor annotations (see Figure 4(top) for some examples from Alexa KB and section 6.1. for evaluation on both datasets). This could be attributed to the large volume of entities and their corresponding denotations in the KBs, which resulted in a number of ambiguous situations. For instance, "*Chicago*" could denote both the **city** and the **broadway musical show**. In the following section we present our new technique of filtering denotations used for Entity Resolution. This method allows our RE system to scale much better than the original method.

3.1. Page-specific gazetteers

We created a new type of entity gazetteer, based on the main entity of a Wikipedia page, and the knowledge about that entity we have in the KB. The new system, presented on the top dashed box in Figure 1, starts with a Wikipedia URL, retrieves its corresponding ID from the KB for that URL (the main entity), and then extracts entities that are connected directly to the main entity (one-hop distance in the KB graph), by going through all the relations the main entity is involved in (except those involving string literals) and returning the entities on the other side of those relations. For each of the related entities, we collect its denotational strings into a purpose-built gazetteer. Figure 2 shows an example KB subgraph for a target entity (in this case George Springate); it contains all the entities immediately connected to it with relations such as **graduate of** or **instance of**. Also appearing in the graph, are the denotation strings for each one of the related entities.

Note that this approach will reduce the number of extracted entities compared to the original method, but will dramatically improve both the coverage for non-NP entities and precision of entity resolution. One way to increase the recall of this system would be to consider entities with a distance of >1 (entities related to entities related to the main entity). Figure 4 (bottom) shows results obtained by performing entity resolution using page-specific gazetteers. Those examples, as well as the results in section 6.1. show that the noise in the data is significantly reduced.

[1] By support sentences we mean any sentence in the dataset that contains both entities.

[2] https://github.com/vered1986/LexNET

[3] https://spacy.io

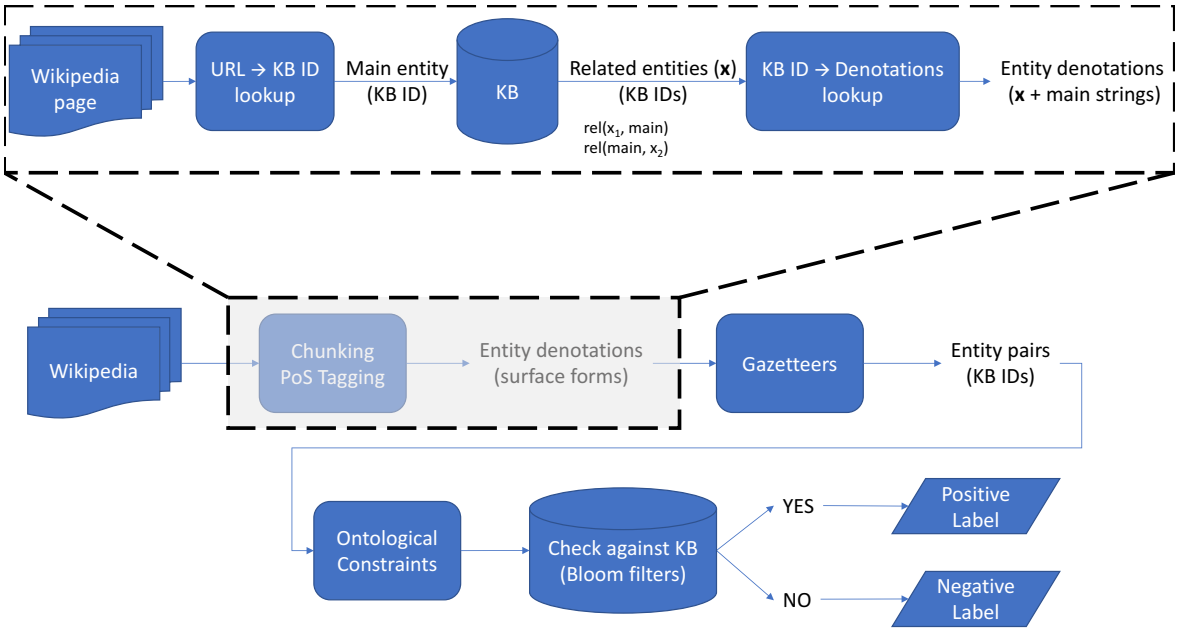

Figure 1: The distant supervision pipeline with page-specific gazetteers. The grey box represents the entity resolution system of Shwartz et al. (2016).

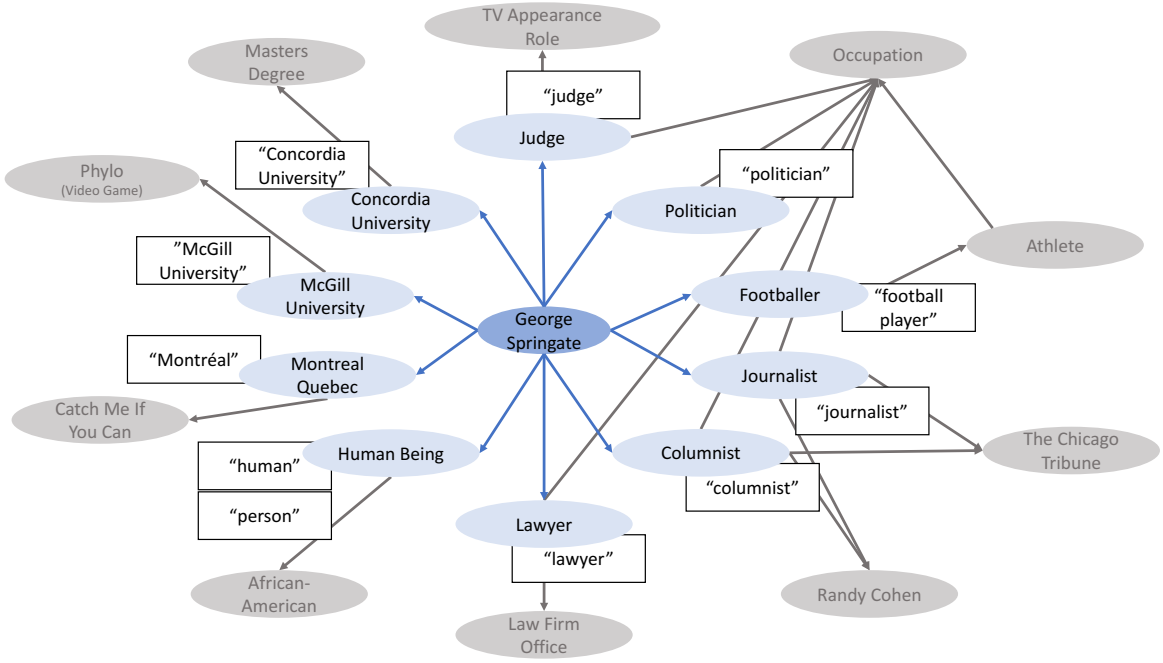

Figure 2: Example KB entry for George Springate. The lightly (blue) shaded ovals represent entities that are within one hop from the main entity and the white boxes are their denotations. The grey ovals represent entities that are two hops away.

3.2. Annotation pipeline

The bottom half of Figure 1 presents the distant supervision generation process adapted from Shwartz et al. (2016) to work with our data. In the original work, the text is processed to split and tokenise the sentences, tag the parts of speech and separate the noun phrases (NPs) – these are the candidate entities. They then construct the dependency path between each possible pair of entities. Each noun/NP pair is checked against the KB for distant supervision. keep only the entities/paths that appear in the list of labelled examples. They also filter out entity pairs that have infrequent paths (occurring fewer than five times), and pairs whose path is more than five tokens long. However, as discussed in the beginning of this section, this approach introduces a lot of noise. To avoid this problem, we use the page-specific gazetteer and a greedy string matching system to scan through the unstructured text and assign KB IDs to the longest-matching substring in a sentence.

The final step was to generate the annotation labels themselves. To do that, we examine each possible pair of entities to see if they participate in the target relation. For the Wikidata KB, we simply checked whether the target relation existed as a property in the data. Considering the large size of Alexa KB, database lookup operations could be very expensive. In order to speed up the lookup for each X rel Y triple, we used two methods. First, before checking against the KB though, we ensure that the pair conforms with the class signature of the relation ('Ontological Constraints'). For example, only a **geographical location** can be the left entity in the **birthplace of** relation. Second, instead of relying on database queries, we used Bloom filters (Bloom, 1970) – a memory efficient probabilistic data structure that can be used to test if an entity is a member of a set. The compression value of a Bloom filter is governed by the accepted false positive rate. We set the false positive rate to 0.001 for our experiments.

Since for any given pair of entities it is much more likely that they are not going to be related, we only keep a small fraction of the negative instances. Following Shwartz et al. (2016), we use a 4:1 negative to positive ratio.

4. Isolating HypeNET features

To discover the effectiveness of the approach of Shwartz et al. (2016), we wanted to separate HypeNET's neural architecture from its input features and use those features with different (and simpler) classifiers. HypeNET's main advantage is that it integrated dependency path features with distributional information about the word lemmas along the path and left and right entities. As our goal was to generate discrete features to be used with more traditional classifiers, we opted for using Brown clusters (Brown et al., 1992) instead of the 50-dimensional GloVe vectors (Pennington et al., 2014) used by Shwartz et al. (2016). The Brown clusters were pre-trained on the Reuters Corpus Vol. 1 (Lewis et al., 2004) using 3,200 clusters.

After evaluating different feature configurations (see section 6.7.), the resulting features were as follows: for each entity pair and for each support, we extracted the dependency path between them and concatenated the lemma, 4-bit prefix of Brown cluster of the lemma, part of speech, dependency relation, and path direction information; to that we added the strings and 4-bit Brown cluster prefix of the left and right entities. The features from different supports were concatenated into one feature list. For example, given the following sentences containing the entity pair **carrie fisher, star wars**: "*In 1977, Fisher starred in George Lucas' film Star Wars*", and "*Fisher became known for playing Princess Leia in the Star Wars film series*". The following is the full list of discrete features extracted, where each space-separated token is a distinct feature, and X and Y are used to replace the left and right entities:

Carrie_Fisher/0111	X/0000/NOUN/nsubj/>
star/0011/VERB/ROOT	in/1101/ADP/prep/<
film/0010/NOUN/pobj/<	Y/0000/NOUN/appos/<
X/0000/NOUN/nsubj/>	become/1111/VERB/ROOT
know/1111/VERB/acomp/<	for/1101/APD/prep/<
play/1111/VERB/pcomp/<	in/1101/APD/prep/<
Y/0000/NOUN/pobj/<	Star_Wars/0011

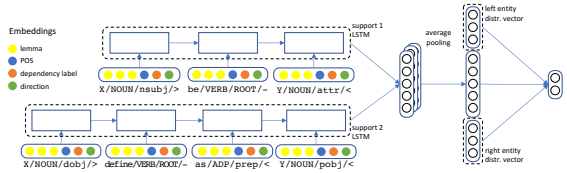

Figure 3: The HypeNET model architecture, reproduced from Shwartz et al. (2016).

4.1. Using a MaxEnt classifier

In the first set of experiments, we used a standard Maximum Entropy classifier from MALLET toolkit (McCallum, 2002) with the discrete features described above. The parameters and settings were kept to their defaults (LBFGS optimizer, with a Gaussian prior variance of 1).

4.2. Using the fastText model

Joulin et al. (2016) recently introduced fastText: a very efficient classifier composed of a simple linear model with a rank constraint. The architecture of the system is very similar to that of Mikolov et al. (2013) except that instead of predicting the middle word in a window, the classifier is predicting a label. For fastText, the input features are token ngrams which are embedded into a single hidden value and fed into a hierarchical softmax classifier. For our experiments, we used fastText's default settings, except for the number of ngrams, which we set to 4.

4.3. Using the HypeNET model

The original version of HypeNET (Figure 3) combines the dependency path-based features with the distributional information in its neural net architecture: for each entity pair, each support (dependency path) token is encoded by a set of embedding layers – one for each linguistic component – and passed into an LSTM layer. The LSTM layers for the whole path are merged by an average pooling layer and the distributional representation of the entities (via embedding layers) is added. Finally, a softmax layer makes a binary classification decision.

We implemented our own version of HypeNET code using Keras (Chollet, 2015) and optimized the learning objective using the Adam optimizer. We modified the basic HypeNET model by making the following changes: i) we allowed the training of word embeddings for lemmas (after initializing them with GloVe embeddings), ii) we replaced the uni-directional LSTM with bi-directional LSTM[4].

5. Evaluation

We want to examine a varied set of connections between the left and right entities, so in addition to the **instance of** relation (P31 in Wikidata) that connects objects to classes, we will examine **birthplace of** (P19) that connects a location entity to a person entity, and **part of** (P527) which links objects to their meronyms. When evaluating against Alexa

[4]The performance of the resulting model was slightly better on the Alexa KB dataset, achieving an F-score value of 94.29±0.21, compared to 94±0.15 for the basic model across the three trials (with a threshold value of 0.5) for **instance of** relation.

His studies were interrupted by army service and at the *end* of the *war* he was forced to return...
instance of (the second world war, cause of death)

In the *intro* to the *song*, Fred Durst makes reference to...
instance of (intro 15367, song)

Turner also released one *album* and several *singles* under the moniker Repeat.
instance of (the singles the 2011 album, album)

Call Your Girlfriend was written by Robyn, Alexander Kronlund and Klas Åhlund, with the latter producing the *song*.
instance of (call your girlfriend 3, song)

Forget Her is a *song* by Jeff Buckley.
instance of (forget her, song)

The *Subei Mongol Autonomous County* is an autonomous *county* within the prefecture-level city of Jiuquan in the northwestern Chinese province of Gansu.
instance of (subei mongol autonomous county, chinese county)

Figure 4: Entity resolution results for the distant supervision training data using Alexa KB and the original pre-processing system of Shwartz et al. (2016) (top), and the new page-specific gazetteers (bottom). The matched strings in the original sentences are *highlighted*.

	Relation	HypeNET μ(F-score)	HypeNET σ	fastText μ(F-score)	fastText σ	MaxEnt F-score
Wikidata	**instance of**	93.90	0.21	**96.44**	0.01	58.45
	birthplace of	92.06	0.90	**93.05**	0.07	66.72
	part of	48.73	2.59	**72.87**	0.16	45.13
Alexa	**instance of**	94.29	0.21	94.31	0.03	83.93
	birthplace of	85.57	0.26	**87.63**	0.01	80.83
	applies to	81.98	1.78	**86.17**	0.01	65.27

Table 1: Mean and standard deviation of the F-score values at 0.5 threshold across three trials for the Wikidata and Alexa KBs, using the MaxEnt, fastText, and HypeNET (our Keras implementation with word embeddings training and bi-directional LSTMs). The MaxEnt model did not have any variance across the trials.

KB, we replaced **part of** with **applies to**, a relation that links an attribute to an object and has no correspondence in Wikidata. We will use the Wikidata KB as a first source of evaluation, and switch to the Alexa KB for a more in depth exploration.

We evaluate all models on a sample of 50K examples for training, 10K examples for validation and test respectively for all relations (except **part of** for which we could only collect 22K training examples). Each example is the collection of all the sentences supporting a $X\ rel\ Y$ triple that have been annotated by the distant supervision system of section 3.. We examine the effect of grouping supports in section 6.4..

6. Results and discussion

We ran each of the following experiments three times (with random initialization) to obtain a measure of variance for their results.

6.1. Distant supervision

The goal of the method presented in section 3.1. was to reduce the number of false positives at the cost of introducing some amount of false negatives (due to missing entities, missing denotations, or missing KB facts). In order to quantify the effect of the new method, we manually annotated 1,000 **instance of** distant supervision examples produced by our new method and the original method used by Shwartz et al. (2016). The original method yielded 67% false positive and 3% false negative examples; the page-specific gazetteer solution returned only 1% false positives and 39% false negatives. After more analysis, 62% of the false negatives (or 24% of the total examples) were cases were the KB contained the **subclass of** relation, which we consider a separate relation (although in the data collected by Shwartz et al. (2016) from Yago and Wikidata it is conflated with **instance of**). The results are similar when using the Wikidata KB: around 1% false positives but only 5% false negatives[5] of which 89% were cases where the KB contained similar relations (like **occupation** for people, or **taxon** for species).

Figure 4 presents a qualitative comparison of the two methods on our KB. We can see that the two problems of spuri-

[5]The lower rate of false negatives can be partially attributed to the exact lookup, instead of the Bloom filters used with Alexa KB.

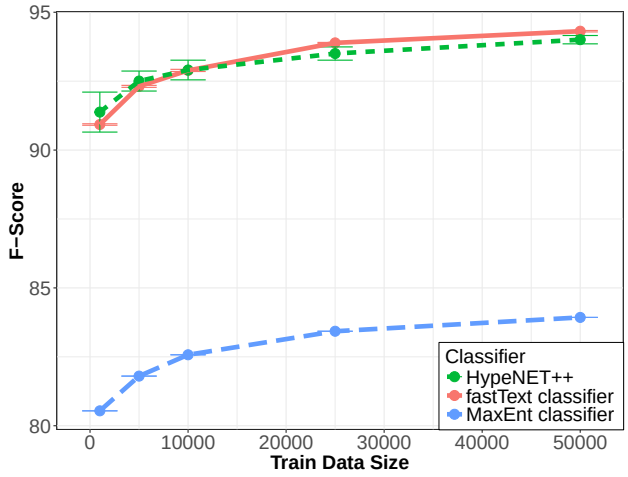

Figure 5: Effect of training data size on the Alexa KB dataset for the **instance of** relation for HypeNET and fastText models at 0.5 confidence threshold.

ous entity matching (e.g. "*end*" to **cause of death**) and non-standard noun-phrase entities (e.g. "*call your girlfriend*") have been successfully addressed by the page-specific gazetteer.

6.2. Model comparison

The results comparing the performance and generalizability of the models (over the three relations) are shown in Table 1. The main takeaway from these results is that the more advanced architecture of HypeNET does not offer a significant advantage over that of fastText when used with (almost) the same input features. As an added benefit, the fastText classifier is dramatically faster than the HypeNET model, with a reduction of training time from around 75 minutes to less than a minute. However as the results of the MaxEnt model show, the features alone are not enough. It is fastText's (and HypeNET's) ability to create higher-dimensional representations of these discrete features that provide the best results.

6.3. Training data size

Another parameter we wanted to explore was the impact of size of the training data since we plan to target relations with fewer training examples in the future. We evaluate the F-scores of the HypeNET, fastText, and MaxEnt models for the **instance of** relation on the Alexa KB dataset.

The results are shown in Figure 5. Note that the numbers in the figure refer to entity pairs, not individual supports (sentences). As expected, the performance of all systems keeps increasing when more training examples are used, but there are two interesting observations to be made. The first is the relative variance of the fastText versus the HypeNET model, especially for the case of fewer than 25k examples. The second is that even with 1,000 training examples, the F-score of both HypeNET and fastText models is above 90%.

6.4. Grouping supports

We also wanted to investigate the effect of grouping the supports (sentences) for each entity pair. As mentioned earlier (Section 2.), this had been proposed as a method to reduce noise in the distant supervision labels (Hoffmann et al., 2011). In Shwartz et al. (2016), the grouping was performed by the mean pooling layer; in the case of the fastText-based system, we simply concatenate the feature tokens from all the supports and feed them into the single hidden layer.

	inst. of	**appl. to**	**bp of**
grouped	94.31	86.17	87.63
ungrouped	93.85	80.90	85.09

Table 2: Effect of grouping supports for each X *rel* Y triple using the fastText classifier on the Alexa KB data (threshold of 0.5).

	inst. of	**appl. to**	**bp of**
satellites	94.31	86.17	87.63
w/o satellites	93.69	85.85	84.42

Table 3: Effect of using dependency path satellite nodes for each X *rel* Y triple using the fastText classifier on the Alexa KB data (threshold of 0.5).

For each of the three relations, we ran the fastText model with and without grouping each entity pair's supports, using exactly the same features in both cases. The Table 2 presents the results. Interestingly, the effect on **instance of** is much smaller than on the other two relations. One possible explanation could be that the page-specific gazetteer method is producing fewer false positives for that relation; more likely, the supports for **birthplace of** and **applies to** are more diverse than those of **instance of**, making their grouping more useful to the classifier.

6.5. Using satellite nodes

We also looked at the role of the dependency path *satellite nodes* (words to the left and right of the entities). This type of features has also been adopted by various systems including Mintz et al. (2009) and Shwartz et al. (2016), and we wanted to establish a basis for its effectiveness across multiple relations. The results, shown in Table 3, show that the clearest advantage of the satellite nodes is for the **birthplace of** relation; for the other two the performance without satellite notes is marginally different. This suggests that the immediate context of the left and right entities is less informative for the **instance of** and **applies to** relations, possible because of the more limited ways that the expression of these relations are syntactically constructed.

	inst. of	**appl. to**	**bp of**
(1) 5 supports	94.55	86.26	87.56
all supports	94.33	85.92	87.63

Table 4: Effect of using all the supports for each X *rel* Y triple using the fastText classifier on the Alexa KB data (threshold of 0.5).

	inst. of	appl. to	bp of
(1)-Brown	94.20	85.93	87.51
(1)-lemma	94.17	84.15	86.65
(1)-POS	94.15	85.93	87.71
(1)-dep	93.59	85.42	86.53
(1)-X/Y entities	93.63	83.89	86.95
X/Y only	91.15	74.20	81.15
full sentence	86.70	77.77	87.09

Table 5: F-score results for the three relations on the Alexa KB dataset. The baseline system (1) is the fastText classifier using the 5 most frequent supports for each X rel Y triple, (1)-dep refers to the system with both the dependency relation and direction features removed, the last system uses all the (lowercased) words in each support as features.

6.6. Using all supports

A comparison is made for the fastText models trained using the 5 most frequent supports for each triple with the ones trained using all available supports. As shown in Table 4, reducing the supports to the most frequent ones slightly increases the performance (except in the case of **birthplace of**) even though on average more than 18K training examples, and more than 3K test examples contain more than 5 supports.

6.7. Feature ablation

As a final step in our exploration, we wanted to measure the impact of each of the features used by the system of Shwartz et al. (2016). Table 5 presents the feature ablation results on the Alexa KB data using the fastText classifier. We compare the full set of features presented in section 4. against feature sets without the Brown clusters, word lemmas, POS tags, dependency information, and the X and Y entities (and their Brown cluster). We also show the results of the system using only the X and Y entities and just the words of the supporting sentences (without extracting the dependency path between entities).

The main takeaway is that for the **instance of** and **applies to** relations, the structure induced by the dependency parser is critical for the system's performance. One explanation is that these relations are not always lexically defined (sometimes expressed with just the verb 'to be' across long subordinate clauses). For the **birthplace of** relation, the system using the full sentence is on par with the best dependency-supported version suggesting that there are strong lexical cues that signify them (like 'born in', or just the presence of a city name).

7. Conclusion

In this paper, we have explored the feature design and network architecture of the HypeNET RE system, and presented a new mechanism for extracting distant supervision data based on our large-scale KB. We found that by replacing HypeNET network architecture with a simple fastText model similar performance is achieved. The main difference between these two architectures is the mechanism of producing the high-dimensional representations: in HypeNET, LSTMs are used, which maintain dependency over longer contexts of dynamic length; in fastText, the window size for the ngrams is fixed. From our experiments, we can infer that dynamic-length context modelling did not bring any gains. Furthermore, we evaluated the effect of grouping of supports and satellites nodes features for various relations. The results from these experiments provide a solid ground to build RE systems for more relations. There are obvious extensions to the current approach, such as using a more sophisticated method for grouping the supports (e.g. an ensemble-based method) and we investigate these in future work.

Beyond architecture improvements, there are two main focus areas for the immediate future: generalising the system to cover very large number ($\sim$1k) of relations, and reducing the sources of noise. The former should be relatively straightforward given the existing architecture for extracting the dataset and training the system. The main obstacle will be to combine the results of the multiple RE systems (one for each relation group) into a single classifier.

Finally, distant supervision as a method itself introduces some errors since not all sentences that mention both entities of a fact express that fact (e.g. the relation **is the director of** between **steven spielberg** and **saving private ryan** is not expressed in the sentence "*The level of violence in Saving Private Ryan makes sense because Spielberg is trying to show ...*"). Going further, we would like to expand the manual annotations to the training/validation sets to assist or replace the distant supervision.

8. Bibliographical References

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. *The semantic web*, pages 722–735.

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM.

Brown, P. F., Desouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Bunescu, R. C. and Mooney, R. J. (2005). A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 724–731. Association for Computational Linguistics.

Chinchor, N., Lewis, D. D., and Hirschman, L. (1993). Evaluating message understanding systems: an analysis of the third message understanding conference (MUC-3). *Computational linguistics*, 19(3):409–449.

Chollet, F. (2015). Keras. `https://github.com/fchollet/keras`.

Craven, M., Kumlien, J., et al. (1999). Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, volume 1999, pages 77–86.

Del Corro, L. and Gemulla, R. (2013). ClausIE: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*, pages 355–366. ACM.

Doddington, G. R., Mitchell, A., Przybocki, M. A., Ramshaw, L. A., Strassel, S., and Weischedel, R. M. (2004). The automatic content extraction (ACE) program-tasks, data, and evaluation. In *LREC*, volume 2, page 1.

Fader, A., Soderland, S., and Etzioni, O. (2011). Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics.

Hoffart, J., Suchanek, F. M., Berberich, K., and Weikum, G. (2013). YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 194:28–61.

Hoffmann, R., Zhang, C., Ling, X., Zettlemoyer, L., and Weld, D. S. (2011). Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics.

Jiang, J. and Zhai, C. (2007). A systematic exploration of the feature space for relation extraction. In *HLT-NAACL*, pages 113–120.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004). RCV1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

McCallum, A. K. (2002). Mallet: A machine learning for language toolkit. `http://mallet.cs.umass.edu`.

McCallum, A. (2005). Information extraction: Distilling structured data from unstructured text. *Queue*, 3(9):48–57.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Mintz, M., Bills, S., Snow, R., and Jurafsky, D. (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: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Parikh, A. P., Poon, H., and Toutanova, K. (2015). Grounded semantic parsing for complex knowledge extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Poon, H., Toutanova, K., and Quirk, C. (2015). Distant supervision for cancer pathway extraction from text. In *Pacific Symposium on Biocomputing*, pages 120–131.

Qian, L., Zhou, G., Kong, F., Zhu, Q., and Qian, P. (2008). Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 697–704. Association for Computational Linguistics.

Ratner, A. J., De Sa, C. M., Wu, S., Selsam, D., and Ré, C. (2016). Data programming: Creating large training sets, quickly. In *Advances in Neural Information Processing Systems*, pages 3567–3575.

Riedel, S., Yao, L., and McCallum, A. (2010). Modeling relations and their mentions without labeled text. *Machine learning and knowledge discovery in databases*, pages 148–163.

Shwartz, V., Goldberg, Y., and Dagan, I. (2016). Improving hypernymy detection with an integrated path-based and distributional method. *arXiv preprint arXiv:1603.06076*.

Varma, P., Yu, R., Iter, D., De Sa, C., and Ré, C. (2017). Socratic learning: Correcting misspecified generative models using discriminative models. *arXiv preprint arXiv:1610.08123*.

Zhou, G., Su, J., Zhang, J., and Zhang, M. (2005). Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 427–434. Association for Computational Linguistics.