Project Proposal

Matt Gardner

I am going to propose one project that I plan on doing with the aid of a 605 student, and two other projects that I could mentor a 605 student (or students) on. I intend to write code and run experiments for the main project, but not for the two mentoring projects. For those two projects, I would provide the resources and guidance necessary for one or two 605 students to be successful on the project, but would not be directly involved in doing the work.

I think all three of these projects, if done well, could easily be submitted to EMNLP at the end of the semester and stand a good chance at getting accepted, either as short papers or long papers, depending on how much of the "further experimentation" is done.

1 Main project: Better features for PRA

My current implementation of PRA selects features based solely on how frequently they connect (source, target) pairs from the training data. This works alright in practice, as there aren't that many features that are seen frequently. But in the near future I want to dramatically increase the size of the feature space I consider with PRA, and what I propose to do here is to come up with better techniques for selecting features in PRA.

A first pass at better feature selection would be to compute precision and recall over the pseudocounts that I generate in the initial random walks, and use F1 or something similar to rank the features, instead of just frequency. A smarter method could select the next best feature given the current features already selected; for instance, if I select a feature that covers some set of training examples with a given precision, I should be less likely to select another feature that covers the same training examples with lower precision, or a feature that covers a subset of the training examples with the same precision. Those are the first two things I plan on trying, and I may come up with other ideas as I am implementing this.

At the same time, to give additional motivation for better feature selection, I propose to add new kinds of features to my PRA implementation, expanding the set of horn clauses that are representable in the feature space. Specifically, I would add one-sided paths, conjunctions of one-sided paths, (possibly) comparisons of one-sided paths, and (also possibly) lexicalized paths. By "one-sided path", I mean a sequence of edges extending from either the source or the target node to any other node in the graph. These kinds of features would allow the classifier to model whether a source or target node is likely to participate in a given relation, in addition to giving a score to the entire triple. By "lexicalized" paths I mean paths that include specific nodes (or sets of nodes), in addition to edge types. Both two-sided and one-sided features could be lexicalized; lexicalizing the path types should significantly decrease the recall of each feature, but in some instances there could be a gain in precision that is worthwhile, particularly with one-sided features.

"Comparisons of one-sided paths" is a little bit more involved. Consider a one-sided path such as <born in >, which for the source entity of a particular entity pair ends at a node "1973", and for the target entity ends at node "1530". A comparison of these dates should be very informative for excluding the entity pair from the relation "married to". So the proposal here is to first find lexicalized, one-side path types, then find some subset of those one-sided paths that can be meaningfully compared, and add a feature to the model that compares them.

Conjunctions of features could conceivably be done by using a kernel function in the classifier. This would alleviate some of the computational stress that conjunctions of features would put on the feature computation step of PRA. There is some work by Arvind Neelakantan at UMass that suggests that combining pairs of

features in PRA's logistic regression classifier gives a significant increase in performance; I would extend this to kernel functions in general, trying kernelized SVMs instead of logistic regression as the classifier. And, if the feature space is large enough, and there is time, factorization machines would be another interesting approach to try.

So, in summary, there are two main parts to this project proposal: better feature selection methods, and more expressive feature generation (where this second part includes some experimentation with the classifier, to get feature conjunctions through a kernel instead of explicit computation). Experiments would be done on the datasets I already have (graphs that have some combination of NELL, Freebase, and SVO relations). In particular, I would pick a subset of Freebase that has numerical attributes (such as cities and countries with populations, GDPs, founding dates, and so on) to use with the one-sided paths, as numerical attributes seem like the best use case for this kind of feature. There is a bit of code that needs to be written to make all of this work, but it should be feasible as a class project by me and another person. I've already spoken with a 605 student who is interested in working on this with me.

2 Proposed mentored projects

2.1 Multilingual PRA

Random walk inference over a graph that connects a formal knowledge base to a large collection of text has been shown to be useful, but these experiments have only been done in English. An interesting experiment would be to see what benefit could be gained by combining several languages into a single graph and performing inference over all of them together. The initial naive baseline would be to just use the SVO triples from each language, connected to Freebase using its multilingual "alias" relation. And then one could experiment with different ways of connecting the SVO triples together, using known translations, or multilingual word embeddings, or some other such thing, to get better joint inference across the multiple languages.

The work involved in this project is the following:

- Pick and obtain a multilingual corpus. Ideally this would be a dataset commonly used for multilingual relation extraction, but my initial cursory search didn't turn up anything useful. A simple fallback if a better dataset can't be found would just be Wikipedia, which has the benefit of having a nice correspondence with Freebase.
- Do a dependency parse of the corpus, for each language. I know there are trained parsers available in multiple languages, but finding them and then running them at scale on a large dataset is the bulk of the engineering work for this project.
- Extract (subject, verb phrase, object) triples from the dependency parsed corpus, in each language. This is the other engineering effort that needs to be done, and is another large data problem.
- Given these sets of SVO triples, pick a set of relations to test on from Freebase, and run my PRA code. This should be quite easy once the data has been obtained. I can help set this up, and provide a machine to run the experiments on, if the students don't have a large enough one available.
- Further experimentation: get a set of translation dictionaries for the languages in the corpus. Either go talk to a friend who works on machine translation and use one of their models, or get find one online (e.g., Wiktionary, or something similar). Use this to add edges to the PRA graph, to allow for inference that spans languages. And in general, just try to see what is the best representation of edges you can get for doing inference jointly over all of the languages.
- Other possible experimentation: get multilingual embeddings to use with my vector space PRA implementation, so that training data from one language can be used to infer new facts in a different language. These embeddings could be obtained from, e.g., Manaal Faruqui's work, or simply from

the SVD implementation in my PRA codebase (the SVD would be performed on all of the languages jointly, hoping to leverage proper nouns to get some overlap among the various languages).

2.2 Sentence-level prediction

My current PRA implementation only makes aggregate predictions over a knowledge base. The same features should be useful for sentence-level relation extraction, however. It should be pretty straightforward to augment a relation extraction model with features generated from random walks over a very large graph. The easiest thing to do here would be to take Hoffmann's model of distantly supervised relation extraction (which has publicly available code), and augment the feature representation used there with features extracted by PRA. This is very similar in spirit to recent work by Jason Weston and colleagues that used a KB embedding to improve performance on Hoffmann's task, and that paper provides a natural point of comparison.

The work involved in this project would be the following:

- Obtain Hoffmann's code and data (paper title: Knowledge-based weak supervision for information extraction of overlapping relations, ACL 2011). The code is downloadable, the data would require emailing him to get it. CMU has a license to the LDC data used, so he will give it to you. It's also possible that someone here at CMU already has it, but I'm not sure. You could alternatively use Surdeanu's model, which is more recent and would provide a stronger baseline; I recommended Hoffmann's simply because I've used it and I'm quite sure this would be easy with his code. It's possible that's also true with Surdeanu's code.
- Get from the data the set of all Freebase entities found in the sentences.
- Use my code to get PRA features between those pairs of entities. There are a few possible ways to do this, all of them would be pretty simple with my code, and I can help.
- Add the PRA features to the features extracted from each sentence in Hoffmann's data.
- Run Hoffmann's code and see how performance changes. Hopefully it goes up, substantially (we're adding a lot of information by giving the algorithm access to the whole of Freebase; there's a problem if performance doesn't go up). The real comparison is against Weston's work: Connecting Language and Knowledge Bases with Embedding Models for Relation Extraction. They also make use of Freebase to bias the relation extraction algorithm, and so the methods are more directly comparable. Which one does better?
- Further experimentation: Take Hoffmann's data and convert it to a graph that can be used by PRA. Run PRA over the joint graph of Freebase and this text, and compare this to the initial experiment. Only the aggregate-level predictions will be comparable, as PRA wouldn't give a sentence-level prediction in this setting, but it would be interesting to see which method gives better aggregate-level predictions from the same data.