CS 512 Assignment 2

Liam Adams

April 23 2019

1 Introduction

The Entity Linking task consists of mention detection and entity disambiguation. Mention detection is the process of identifying the named entity and entity disambiguation consists of using the context surrounding the entity mention to determine which specific entity among entities with the same name is relevant. For example, the entity mention "Michael Jordan" could be the famous basketball player or a famous professor from UC Berkeley. After identifying and disambiguating entity mentions, the final step is to link them to their corresponding entry in a knowledge base.

Entity linking is an important task for semantic text understanding and information extraction. This is useful for downstream Natural Language Processing tasks such as automatic knowledge base construction, text summarization, and relation extraction. EL can also assist in assessing the type information of entities, and in question answering systems. Roughly speaking, EL can be used in a QA system by decomposing a question into its entity mentions, and using information from the links to a KB to find promising information to produce an answer.

My model was inspired by (Mueller and Durrett, 2018). The major contribution this paper aims to make is to more effectively use the context surrounding entity mentions to produce better quality links. I thought this paper was especially relevant to our task since we were given the left and right context for each entity mention in our training data. The paper observes that the state of the art attention based models fails to identify some of the most useful context words surrounding an entity.

It also aims to improve the method for training a model. Like previous works and this assignment, the training data consists of a list of candidate titles that have been previously linked to the entity mention, and the gold title which is the correct candidate title for the entity mention. However instead of making this a binary classification problem (did the model predict gold or not?), the training is structured as a multi class decision among all the candidate titles while maximizing the probability that the gold title is chosen.

In contrast to previous works, they used GRU input layers in their neural network rather than LSTM.

In addition to the GRU layers they also use a novel scheme for attention over

the context based on the title. It allows the model to weight the importance of the output of the GRUs at each time step. The left and right contexts each have their own attention weights. This method enables the model to focus on the more relevant context (left or right). However this alone did not enable the model to use the most relevant individual words because some of the indicative words do not contain enough common information in their embeddings with the entity mentions, which precludes them from being deemed relevant.
Their first attempt at designing a method to use these more indicative context words used character level information fed into a convolutional neural network. They found this still did not identify more relevant context words, but they did devise a method using sparse features to target the relevant context words directly. These features used character level information in the form of identifying context words that were exact matches with the word(s) in the entity mention, or a substring of word(s) in the entity mention. These features are then appended to the left and right context vectors after going through the GRU and attention layers before passing through the final layer to produce the output.
One of the drawbacks of this method is that context words that match or are substrings of the entity mentions are not always disambiguating terms, so they may produce false positives. Also these character matching features they introduced to boost the performance do not learn any deeper semantic meaning in regard to the entity mention.

2 Methodology

My methodology uses a neural network with a GRU input layer, a hidden layer, and a final sigmoid activation function to produce an output between 0 and 1. I treat this as a binary classification problem so that each candidate is classified as ground truth or not ground truth. I chose a GRU input layer because this was the choice of Mueller and Durrett. GRUs seem to be good for sequential textual data in which you have a word or entity mention surrounded by some contextual data. Figure 1 below shows a diagram of the architecture.
Each candidate in the training set is a training sample. For each candidate I use the embedding data provided with the project to look up the embedding vector for the entity name v_e and the embedding vectors for the context words. I sum up the vectors for the context words to get 1 vector v_c, combining both the left and right context. I then build a vector v_h with some hand crafted features. The features in this vector are the similarity between the candidate name and the surface name using Python's $SequenceMatcher$, the prior probability provided for each candidate, and cosine similarity between v_e and v_c. These 3 vectors a concatenated to form a row vector with 603 columns. The GRU requires a 3 dimensional input (seq_len, batch_size, num_features), I use a batch_size of 32 and 1 for seq_len as each candidate is evaluated independently. The GRU then transforms this into a hidden layer (batch_size x 100) which is fed to a linear transformation to obtain a column vector with $batch\_size$ rows. This column

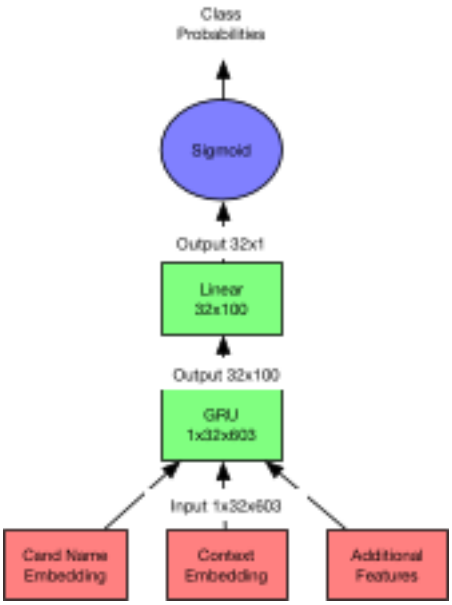

Figure 1: Neural Network Architecture

vector is then fed to a sigmoid activation function to produce values between 0 and 1 for each candidate in the batch.
I use the Adam optimizer and the Binary Cross Entropy loss function. After each batch the loss function calculates the loss for that particular batch and propagates it through the modules in the network, then the the optimizer updates its parameters for this new state. I use a batch size of 32 and 50 epochs, so that the entire training set is fed to the network 50 times.
When making predictions for the test set, I create my 3 dimensional input tensor to provide the GRU. However instead of using a batch_size of 32, my batch size is the number of candidates for the current test entity mention. I feed that to the network and get a number between 0 and 1 for each candidate. I treat these values as the probability the candidate belongs to class 1 (is the ground truth), and choose the candidate with the highest value as my prediction for the ground truth.
During training I log the loss for each batch, with each batch starting with a loss of 0. For the first several epochs the loss decreases dramatically before seeming to converge around 50 epochs. It ends up settling around .09 or .1.

3 Results

Below is a table of F1 scores for each model on each test data set. For the most part, each step improved the F1 score of each dataset.

	Aida A	Aida B	MSNBC	ace	aquaint
Random	.3590	.3382	.3963	.2777	.3175
Prior	.7383	.7191	.8982	.8732	.8448
Supervised	.7427	.7220	.8982	.8732	.8601
Embed	.8352	.7958	.9135	.8853	.8853
Neural	.8632	.8314	.9151	.8813	.8881

Table 1: F1 Scores

Below are a few examples in which my model predicted the wrong candidate as the link. First we have

"Grace Road Leicestershire extended their first innings by 94 runs before being bowled out for 296 with **England** discard Andy Caddick taking three for 83 Trailing by 213 Somerset got a solid start to their second innings"

The entity mention is in bold, the text to the left of the entity mention is a subset of the left context and the text to the right is a subset of the right context. My model predicted *England* as the link when the correct link was *England cricket team*. I believe my model got this wrong because the prior probability of England here is .827, which is much higher than any other candidate in this case. Also England exactly matches the entity mention. Two features my model uses are the prior probability and similarity between the entity mention and candidate names, so I believe the strong signal from these two features is why the prediction is wrong, even though the context is clearly talking about sports.

Another example is

"... Such had scuttled their hopes taking four for 24 in 48 balls and leaving them hanging on 119 for five and praying for rain At the **Oval** Surrey captain Chris Lewis ..."

My model predicted *Adelaide Oval* as the link when the correct link was *The Oval*. The Oval actually had a higher prior probability, .115, than Adelaide Oval at .08. This sentence is talking about an English cricket match and the entity mention is the name of a cricket ground in London. Adelaide Oval is a cricket ground in Australia. The right context goes on a bit more and mentions an Australian player playing for one of the English teams. I believe my model got this wrong because there is some ambiguity in the context with mentions to England and Australia, and the model may not have considered "the" preceding Oval to be a discriminant term, when in fact it was in this case.

Here is one more example

"The Iranian army announced today Saturday that it had successfully completed a new test on Shahab 3 ground ground missiles An official at the Iranian **Ministry of Defense** told Tehran Radio that the Shahab 3 defense missile was tested a second time"

My model predicted *Ministry of Defence (United Kingdom)* when the correct link was *Ministry of Defense and Armed Forces Logistics (Iran)*. The prior probability of my model's prediction was only .13, not the highest among the candidates, however the prior probability of the correct link was .005. I believe my model predicted wrong because this is such a low probability it must have precluded it from being selected. Even though the context mentions Iran several times it appears the low probability outweighed the context.

In general most of the incorrect predictions my model made were for candidates having a very high probability and similar name to the entity mention. These specific features seemed to outweigh the more abstract context features from the word embeddings. Also there were some incorrect predictions simply because the candidate list was empty.