|  |
| --- |
| **Comparison of Named Entity Recognition**  **Systems for Trivia Questions** |
| [**https://github.com/tonycolucci/MSIA414\_FinalProject**](https://github.com/tonycolucci/MSIA414_FinalProject) |
| **Anthony Colucci**  **Northwestern University**  **anthonycolucci2014@u.northwestern.edu** |
| 050  051  052  053  054  055  056  057  058  059  060  061  062  063  064  065  066  067  068  069  070  071  072  073  074  075  076  077  078  079  080  081  082  083  084  085  086  087  088  089  090  091  092  093  094  095  097  098  099 |
|  |
|  |
|  |

Abstract

|  |
| --- |
| 000  001  002  003  004  005  006  007  008  009  010  011  012  013  014  015  016  017  018  019  020  021  022  023  024  025  026  027  028  029  030  031  032  033  034  035  036  037  038  039  040  041  042  043  044  045  046  047  048  049 |

This paper examines the interaction of Named Entity Recognition and Question Answering through a survey of existing Named Entity Recognition libraries and an experiment using a set of 200,000 questions from the popular quiz show Jeopardy. The primary focus of this paper was to examine different implementations of Named Entity Recognition through widely available libraries spaCy and StanfordNER. Following this an experiment is run documenting the effects of a small amount of additional training for the pre-trained spaCy NER model to identify potential improvements from training data that better matches the format of questions and answers. However, this additional training did not improve results over the test set of data, decreasing NER performance by 0.4-3%.

Introduction

Question answering is both a fundamental task of Natural Language Processing (NLP) and of answering trivia questions. A machine tasked with understanding a question as one would appear on Jeopardy would be required to pick out people, places and events as they are typically the primary indicator of an answer to a human answering the same question. While a machine certainly goes through different processes, Named Entities will still typically contain essential information for answering trivia questions. In this paper, we introduce a set of Jeopardy-style questions to an existing Named Entity Recognition model to see if training on a set of data with this specific format may offer gains in performance over the baseline model trained over other standard sets of text data.

Literature Review

**Previous Research:** Named Entity Recognition has been one of the foundational tasks of Natural Language Processing since it’s recognition by MUC-6 in 1995. Named Entity Recognition plays a role in both the Question Processing and Answer extraction elements of answering a trivia question. The most common task for NER is performance on a dataset compiled and released at the 2003 Conference on Natural Language Learning (CoNLL). Molla, Van Zaanen and Smith (2009 examined how NER can improve results on a Question Answering task, focusing on the ability of NER to narrow down a set of potential answers to the most likely possible answer. Lample and others at Carnegie Mellon University published a paper in 2016 introduced the architectures of bi-directional LSTMs and Conditional Random Fields for tagging as a new state-of-the-art in the field. The current state of the art was recently achieved by a team at Facebook (Baevski, et. al., 2019) using bi-directional transformers.

**Library Comparison:** When attempting to implement an NER solution, the most popular methods to do so are spaCy, which is a Python library with powerful, fast out-of-the-box NER power and StanfordNER, implemented in Java, which has been around for much longer. Considering first StanfordNER, (Finkel, Grenager and Manning, 2005) this implementation uses Conditional Random Fields and, at the time of publication, represented near state-of-the art performance in the 2003 CoNLL task. This model makes use of bi-directional context, character-level n-grams and POS tagging to represent sequences which are then tested using methods similar to Markov Chain Monte Carlo Methods.

The spaCy module (Honnibal & Montani, 2017) is more easily usable than the StanfordNER model, but has less published material regarding the techniques behind it and its level of performance, which was near state-of-the-art at its much later introduction. The spaCy NER models make use of Bloom word embeddings, which allow for compression of data representation in an efficient manner (Serra & Karatzoglou, 2017) and residual Convolutional Neural Networks in generating the entities.

1. Data

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

|  |
| --- |
|  |

This project was working with the dataset consisting of all questions and answers from the quiz show Jeopardy between 1985 and 2012. This dataset covers 216,930 data points, each of which is has an associated show date, as well as a category, question and answer, each of which are text data. Each of the question texts are posed as a statement, as opposed to a question, and the answers are typically single words or entities. It is important to note that answers are not phrased as questions in this data, though the rules of the show demand that contestants respond in the form of a question. This data is not labelled for any standard NER tasks, so any labeling for the experiment would need to be done by hand. Initial observations were done over this set of data including generating word2vec embeddings of questions and answers and

1. Experiment

**Process:** This experiment went through several stages of failure before reaching the results described here. I first intended to compare implementations of NER across libraries and to examine the effects of adding substantial training data in the form of the Jeopardy data set. Unfortunately, the lack of labels on the Jeopardy dataset posed the first problem. I attempted to go through a hand-labeling process, but this would not provide a significant scale for the data. I additionally encountered trouble in setting up a local version of StanfordNER due to issues creating a local environment in which to run Java scripts. I also considered running an experiment around word embeddings of question texts versus answer texts to see if cosine similarity may be identified among certain sets of words of the question and the answer, but decided upon the experiment described below.

**Experiment Design:** My experiment examined the effect of a small amount of Jeopardy-style training data in addition to the larger pre-trained spaCy model. do so only allowed for 50 training data points and 50 test data points to be labelled. The baseline for the experiment was the out-of-the-box spaCy NER implementation, labelled as baseline. This model was then additionally trained over a sub-sample of the 50 labelled training observations, establishing a model train\_38. The baseline model was then trained over all of the labelled training observations, creating the model trained\_50. All additional training was done using the resume\_training function of the spaCy module which allows for the model to take in more observations while not losing the context of the baseline model. These models were then each tested against the labelled test observations.

1. Results

As shown in Figure 1, the additional training data did not, on aggregate, increase the predictive power of the model. The train\_50 model performed very near to the baseline model, while the train\_38 model saw a more significant dip in performance according to accuracy. Accuracy was an appropriate metric in this case because we did not have evidence of particularly unbalanced classes in the data. In fact, most of the NER results did not change across the models, which is not particularly surprising for the small amount of additional training. The primary conclusion from this experiment is that more labelled data or a more significant change of methodology would be necessary to improve on the baseline model, even when introducing a new format of text as we are with trivia questions.

|  |
| --- |
|  |
| Figure 1: Relative Accuracy of the baseline models and those**.** |

Acknowledgments

I would like to acknowledge Emilia Apostolova and Timo Wang for providing the instruction for this class and essential context of how this experiment and these concepts fit within the wider subject of NLP.

References

Baevski, A., Edunov, S., Liu, Y., Zettlemoyer, L. and Auli, M., 2019. Cloze-driven pretraining of self-attention networks. *arXiv preprint arXiv: 1903.07785*.

Honnibal, Matthew and Montani, Ines. 2017. <https://spacy.io/> spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005),* pp. 363-370. <http://nlp.stanford.edu/~manning/papers/gibbscrf3.pdf>

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition.

Molla Aliod, Diego & Zaanen, Menno & Smith, Daniel. (2009). Named entity recognition for question answering. 51-58.

Serra, Joan and Karatzoglou, Alexandros. 2017. Getting Deep Recommenders Fit: Bloom Embeddings for Sparse Binary Input/Output Networks