Exploring Neural Architectures for NER

CS224N - Project Milestone

Team

- Vincent Billaut, `vbillaut`, 06225465, MS (Statistics)
- Marc Thibault, `marcthib`, 06227968, MS (ICME)

Mentor

Richard Socher

Problem description

Our goal is to tackle the problem of **Named Entity Recognition** (NER) using neural architectures. More precisely, we ought to reproduce, and try to carefully study the protocol described in [1]. The added value we wish to bring notably consists in a detailed study of **classification errors**, in order to better understand where this approach *fails*, and where/how it can be *improved*.

Data

We are using the same dataset as the one used in [1], namely **CoNLL-2002** (Tjong Kim Sang, 2002), and will potentially consider using CoNLL-2003 ([4]) later on, if this proves relevant.

The dataset consists in **1.05M** labeled words (**150MB** csv file).

Baseline and first models

Our baseline model is a very simple **feed-forward neural network** with one hidden layer. It takes the inputs word by word, uses a GloVe word embedding and optimizes for cross entropy loss.

Its main objective is to provide a sane first architecture to test our code pipeline and the different functionalities with have implemented to ease the benchmarking and development of more complex algorithms (shell argument parsing, base NERModel and Config classes that the other implementations will inherit from, ...). This was also a way to have some script to test our GPU configuration on, which we did successfully.

In addition to this baseline, we have implemented two other models: a **simple LSTM**, and a **variation** of it. The simple LSTM consists in a basic LSTM cell, the output of which is directly taken as the prediction for our labels. The variation of this model is to simply add **an extra layer** on top, so that there is some "learning room" between the cell and the output layer. So far, on very small sets, the LSTM models don't perform well, and we're waiting for our larger runs on the whole dataset to see actual results, and to decide how to tune those hyper-parameters.

Evaluation

Our evaluation, much like in assignment 3, will be based on entity-level F_1 score, on a separated development set. Our goal will next be to examine some of the misclassified labels, and to conduct a more thorough error cases study.

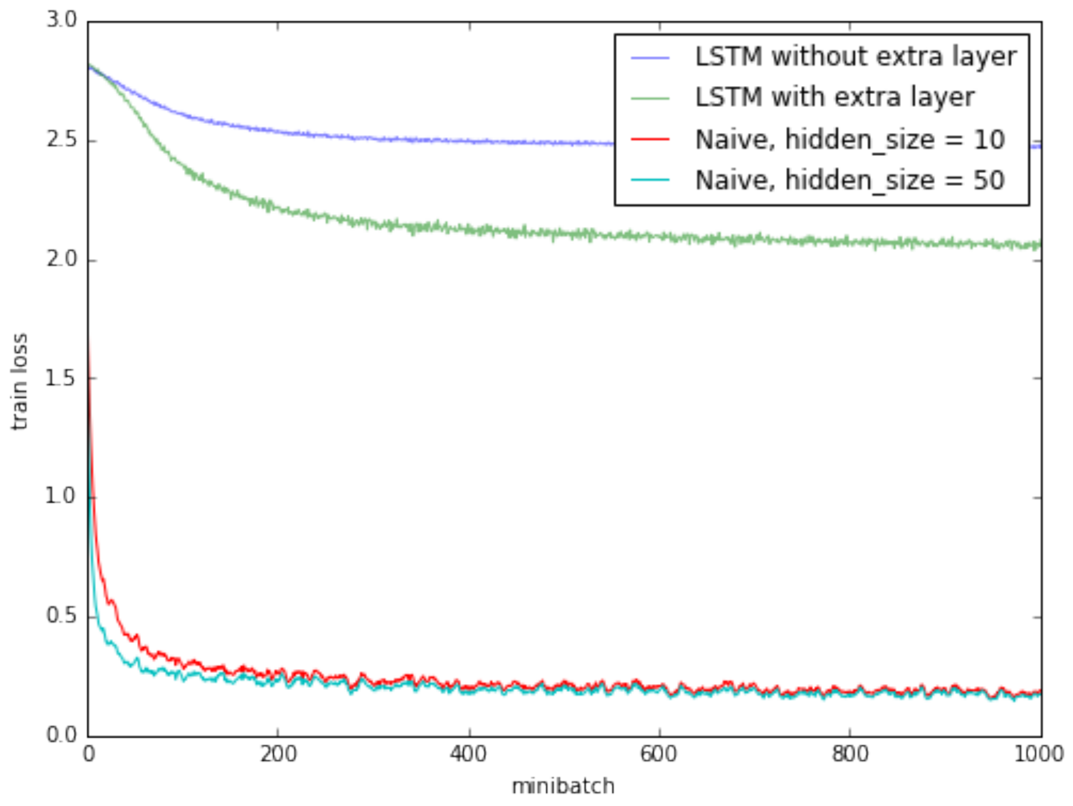
So far, our baseline (`Naive`) model ends up with a dev F_1 score around 0.7, which is very decent for this task regarding how simple it is, but very far from what we aim to achieve (the main papers we consider hover around 0.9 scores on this dataset).

Results

Model	**CE loss**	**Entity-level F_1 score**
`Naive, hidden_size = 10`	0.16	$.69$
`Naive, hidden_size = 50`	0.14	$.69$
`LSTM`	2.44	$.08$
`LSTM + extra_layer`	2.01	$.23$
`LSTM + extra_layer + extra_epochs`	1.96	$.60$

From the most recent models we fitted, we see that the baseline works best both in train CE loss; as well as in dev entity-level F_1 score.

However, the encouraging result is that, by pushing our LSTM model further into the epochs, we see an explosion in the dev set F_1 score. This means that the LSTM still has room for improvement, and might be more accurate than and eventually outperform the baseline with longer runs.



Next steps

We have begun working on other model implementations, such as **Bi-directional LSTM** and **LSTM-CRF** (Conditional Random Fields, see [1]), and while the former will probably not prove challenging to implement, the latter will definitely require some intense work.

Bi-directional LSTM is really the baseline we want to work off of, since it is straightforward to implement from what we have already done, and it has become the vanilla model for NER tasks.

Related Work

This part comes directly from the project proposal, as we thought it was relevant to mention the main articles we use again, here (even the ones we don't specifically mention above).

> **[1]** Lample, Guillaume, et al. "Neural architectures for named entity recognition." *arXiv preprint arXiv:1603.01360* (2016).

This article is the basis that will guide our work. It uses LSTM-CRF models and compares them to transition-based approaches.

> **[2]** Collobert, Ronan, et al. "Natural language processing (almost) from scratch." *Journal of Machine Learning Research* 12.Aug (2011): 2493-2537.

This article implements CNN methods for NER.

> **[3]** Huang, Zhiheng, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF models for sequence tagging." *arXiv preprint arXiv:1508.01991* (2015).

This article implements LSTM models very similar to the first approach described in [1], and will prove useful in implementing the neural architecture ourselves. It notably shows how to combine LSTM with CRF.

> **[4]** Tjong Kim Sang, Erik F., and Fien De Meulder. "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition." *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, 2003.

This article describes very generally the task of named entity recognition, and provides a common dataset which has been used as a standard for NER ever since, and notably by the aforementioned articles.