Natural Language Processing (almost) from Scratch

1 Idea

The paper attempts to train a generic single learning system for multi-task learning. The tasks include Part-of-Speech (POS) tagging, chunking (CHUNK), Named Entity Recognition (NER) and Semantic Role Labeling (SRL). The authors intend to achieve this without hand-engineering task-specific features, and instead rely on a large amount on unlabeled data. They also wish to avoid baselines that have been created using differently labeled data.

2 Background

- The state-of-the-art (SoTA) system for POS tagging uses bidirectional sequence decoders (Viterbi algorithm) and maximum entropy classifiers to determine, which among a set of pre-defined tags, can be attributed to a token.
- Chunking is essentially the same as POS-tagging, but for phrases instead of single words. SoTA for chunking uses pairwise SVM-classifiers, for which the features were word-contexts. Matrix SVD based methods have also been successful.
- For NER, the SoTA is a linear model combined with Viterbi decoding, where the features include the tokens themselves, the POS tags, CHUNK tags, suffixes and prefixes.

- SRL is similar to obtaining an entity-relation model from unstructured data (text). SoTA on SRL are parse trees, CHUNK and POS tags, voice, types of verb etc. in combination with context-window classifiers.

3 Method

- The system used by the authors is a simple MLP architecture with minimal pre-processing and no task-specific engineered features.
- The first layer extracts word-level features and the second layer extracts sentence-level features. The architecture uses the equivalent of an embedding/lookup layer that models the language by learning dense representations of words. Initial embedding layer size was 50 (increased to 500 in later experiments). Vocab size varied from 100k to 130k words. Sentence level embedding is learnt by using convolutions.
- The authors also propose training a small network, and using the trained embeddings to initialize a larger network, as a form of transfer learning.
- There could be multiple lookup tables, with the feature vector for a word being the concatenation of all the lookup tables entries.
- For some tasks, the training objective is a multi-class softmax probability and for others, the objective is to collectively maximize the probability of the entire sequence rather that the prediction at each step individually.
- In contrast to previous methods, the authors use the *tanh* non-linearity as the activation function in their neural network architecture. The authors also explain that not have non-linear activations in a multi-layer architecture is the same as having only a single layer.
- For convolutions, the tag prediction is done for the word in the middle of the convolutional window. Padding is done to ensure that there are tokens preceding the actual first word, and tokens following the last word.
- The training objective is the maximization of the log-likelihood and the optimizer used is stochastic gradient ascent.

- Sequence-like decoding for each of the tasks is done using the Viterbi algorithm that uses dynamic programming to maximize the likelihood of entire sequences.

- The proposed system eliminates the need of parse-trees for the SRL task. The authors hint at this being a deviation from Chomsky grammars which is hierarchical, and describe this as a Harris grammar which is similar to a set of functions being applied on top of a sequence.

- The authors also point to how features from different tasks can be combined into a smaller subspace, and an two objectives could be alternatively trained, one for the meta learner, and one for the task specific learners.

- The authors implemented the neural network from scratch in C, without a symbolic computation framework. This is presumably a very time consuming effort since every back-propagated gradient would have to be hand-computed first, and then coded.

4 Observations

- One pertinent question is whether multi-task learning is only effective if the subtasks are not completely orthogonal to each other. This is not evident from the paper's conclusions because a few of the subtasks are dependent on features extracted from the other tasks.

- The network displays consistently good performance across all tasks, sometimes even surpassing the baselines, especially when provided with additional data, and by using transferred embeddings.