This paper presents neural network models which are able to perform NLP tasks “(almost) from scratch”. Thanks to the ability to discover adequate internal representations, these neural network models avoid ask-specific engineering and excels on multiple benchmarks including part-of-speech tagging (POS), chunking (CHUNK), named entity recognition (NER) and semantic role labeling (SRL).

The neural networks sequentially contain a lookup table, a linear layer, a hardTanh layer and another linear layer, when the input is a window of a sentence. In contrast, given a whole sentence as input, before entering the first linear layer, *local* features are extracted for each word in the sentence with window approach, and these features pass a convolution layer to generate a *global* feature vector, and then a max layer (not average layer) to capture the most useful local feature. The former window approach works well for POS, CHUNK, NER, while the latter sentence approach is more suitable for SRL.

The training is implemented by maximize a likelihood using stochastic gradient ascent. There are 2 types of log-likelihood being utilized: word-level and sentence-level. Given a word and a tag, the word-level log-likelihood is similar to regular softmax, which is the score of the label normalized by the “logadd” of scores of all possible labels. For sentence-level log-likelihood, similar concept of score is introduced for a sentence and a tag sequence pair. However, in order to taking into account the sentence structure, the score here takes in all word-level scores for word-tag pairs along with transition scores A’s representing jumping of tags in successive words. Thanks to associativity and distributivity of “logadd” on the semi-ring, by using a recursive technique (with memorization?), the time complexity of computing “logadd” over all choices of tag sequences given a sentence could be optimized to linear time from exponential time with respect to the sentence length t. Moreover, at reference time, by replacing “logadd” with “max” operation, Viterbi algorithm could be conveniently applied.

The supervised benchmark results are not that satisfying, and the authors find that in the word embedding space/lookup table, semantically similar are not close as assumed. Since most trainable parameters are associated with word embeddings, they decide to use much more training data to obtain better ones. With the belief that words corresponding to similar tags appear together in text and that this correlation could be captured by the likelihood of observing a word given the context consisting of “similar” words, a lot of unlabeled data (from Wikipedia and Reuters) is taken as the input of the proposed neural network model to train a language model by stochastic gradient minimizing the ranking criterion of a stochastic gradient. Here the ranking criterion is favored by the authors since the cross entropy criterion, greatly affected by most frequent phrases, lacks “dynamic range”.

Then semi-supervised benchmarks are discussed. For “semi-supervised”, the word lookup tables are initialized with the embeddings computed by the language models using abundant unlabeled data, and they are free to modify during supervised training stage. At this time, the results are competiable to benchmark results.

Then the authors consider other means for further improvements. Multi-task learning is such a possible way. Models for related tasks of interests are jointly trained with an additional linkage between their trainable parameters in the hope of improving the generalization error. A simple approach for joint training is to have shared parameters among different models. (In contrast, joint decoding considers additional probabilistic dependency paths between models, which therefore defines an implicit supermodel describing all tasks in the same probabilistic framework. Without joint training, the additional dependency paths cannot directly involve unobserved variables. Thus, it is natural to use joint training for discovering common internal representations across different tasks.) In the unified model trained by the authors, word lookup table, first linear layer and part of convolution layer are shared for models of different tasks. The training is achieved by minimizing the loss averaged across all tasks. There are 2 observations: it produces a single unified network that performs well for all these tasks (sentence approach); while it only leads to marginal improvements over using a separate network for each task.

Moreover, popular techniques for task-specific engineering including suffix features, gazetteers, cascading, ensembles, parsing, word representations are reviewed, added to the neural network models and compared with original models without task-specific engineering.

Finally, the authors conclude that the proposed models avoid task-specific engineering as much as possible. Being able to discover internal presentation with the neural network, they could achieve competitive results compared to benchmarks in different NLP tasks. However, there is no free lunch, the feeding of word embedding from a long-time-trained language model is needed for best performance.