This paper presents a dependency parser using neural network classifiers. Conventional feature-based discriminative dependency parsers have some drawbacks: (1) use of many mainly poorly estimated feature weights, (2) dependency on manually designed set of feature templates (expertise and incompleteness), and (3) time-consuming feature extraction steps. In this paper, the use of dense features addresses these problems by *sharing* *statistical* strength between similar words. Furthermore, in order to encode available information for the configuration and to model higher-order features based on the dense representations, the authors train a neural network classifier for parsing decisions (encoding?) within a transition-based dependency parser (modeling?).

For transition-based dependency parsing, the arc-standard system is employed. A configuration consists of a stack s, a buffer b and a set of dependency arcs A. There are 3 types of transitions: LEFT-ARC(l), RIGHT-ARC(l) and SHIFT. The goal of this greedy parser id to predict correct transitions based on given configuration (which contains information including POS tags, word head and label, position of words in stack/buffer).

The neural network architecture contains input layer, hidden layer and softmax layer. A set of elements based on the stack/buffer positions for each type of information (*word/POS tag/label*) are chosen. Embeddings for chosen words/POS tags/labels are added into the input layer. Then, the input layer is mapped to hidden layer through a *cube* activation function. Finally, the softmax layer is used to model the multi-class probabilities. It is worth mentioning that semantic meanings of word/POS tag/label are expected to be captured by the dense representations. Moreover, the cube activation function is believed to better capture the interaction of three elements.

Experiments on different languages, different metrics with different dependency representations shows that the proposed model gives better accuracy and speed. There are some interesting details: (1) cube activation function is suitable for this task, (2) initialization of pre-trained word embeddings outperforms random ones, and (3) POS embeddings might contain most information of label embeddings.

Further model analysis shows that (1) embeddings for POS tags and labels could carry some semantic information (similarities for close embeddings, regularities), (2) POS tags are important in dependency parsing, and (3) weight matrix in hidden layer are able to *automatically* capture features (e.g., plausible features with large weights).