SVM Model:

For this task we used the Soft-SVM model from the sklearn library with the RBF (Gaussian) kernel.

We started by naively loading the entire train set, computing the “glove-twitter-300” embeddings for each word in the train set and fitting the SVM model on it.

However, this process took too long, so we decided to filter the dataset, since the vast majority of words are tagged ‘O’.

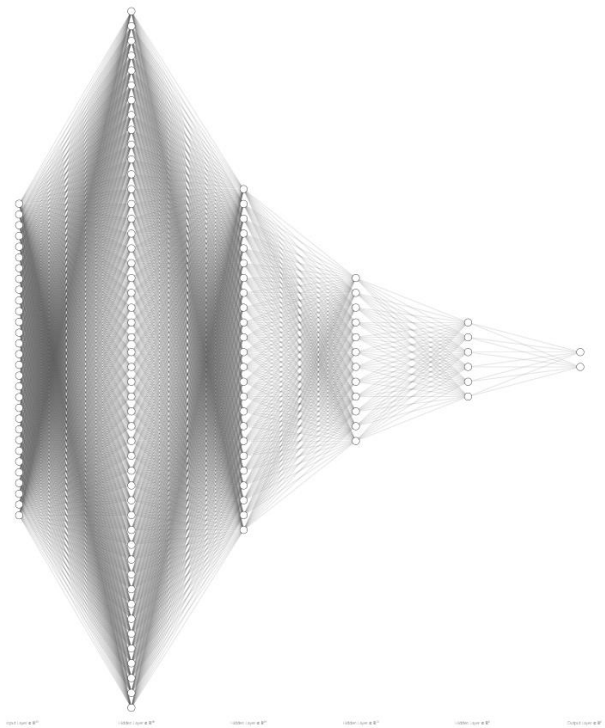
Therefore, we took the first 12000 negatively labeled words and all the positively labeled ones to train the model, achieving an F1 score of ≈ 0.6 on the tagged development file.

FF Model:

The model used in this section was a fully connected neural network (MLP).

The architecture of the model is the following:

- Input size – 300
- Hidden1 size – 2048
- Hidden2 size – 1024
- Hidden3 size – 512
- Hidden4 size - 256
- Output size – 2
- Batch size - 256



Preprocessing:

As in the SVM model, we compute the Glove embeddings of each word in the train set, only this time over the entire train set. Next, we compute the contextual embeddings of each word by concatenating the embeddings of each word between it’s previous and next words, where the word that comes before the first word in a sentence or after the last one is defined as “End_Of_Sentence” and given the zero-vector embedding.

In addition, we compute the ratio of positive to negative instances in the data since there is a significant bias towards negative labels and use this ratio when updating the model.

To train the model we used weighted binary cross-entropy loss, with the ratio computed before used as the weight for the negative class.

The optimizer used during training was AdamW.

With this model we achieved an F1 score of 0.615 on the development set.

Competitive Model:

The model we used in this section was a bidirectional LSTM followed by a shallow fully connected network to decode the LSTM output for each time step.

In order to get the best results with the LSTM model, we pass each sentence to the model as a batch in itself, but preform backpropogation every 64 sentences.

The loss used here is Cross Entropy loss, just like in the feed-forward model, where the output of the model is a vector of probabilities for each class for each word.

The word embeddings used here are the same as in the SVM model, with no contextual embedding.