## Bonus Tasks

The maximum grade you can get for the assignment is an 8. If you want to obtain a better grade, you need to individually send results for one of the bonus tasks to intro2nlp@googlegroups.com. If the group project grade is less than an 8, we do not check the bonus task submission. If the group project grade is an 8 and you submitted an answer for a bonus task, you might still only receive an 8, if the quality of the bonus task submission is not sufficient.

Task options:

* Provide answers for exercises 8 and 12-14 for at least one of the other languages of the CWI task.
* Improve the model by making a substantial change. Varying a hyperparameter or simply adding another layer **is not** a substantial change. Motivate your modification and interpret the findings.
* Identifying complex words is only the first step for lexical simplification. Read up on related work and explain potential architectures for contextualized lexical simplification in detail.

For my submission I chose the second option of improving the model (found in net.py in the model directory):

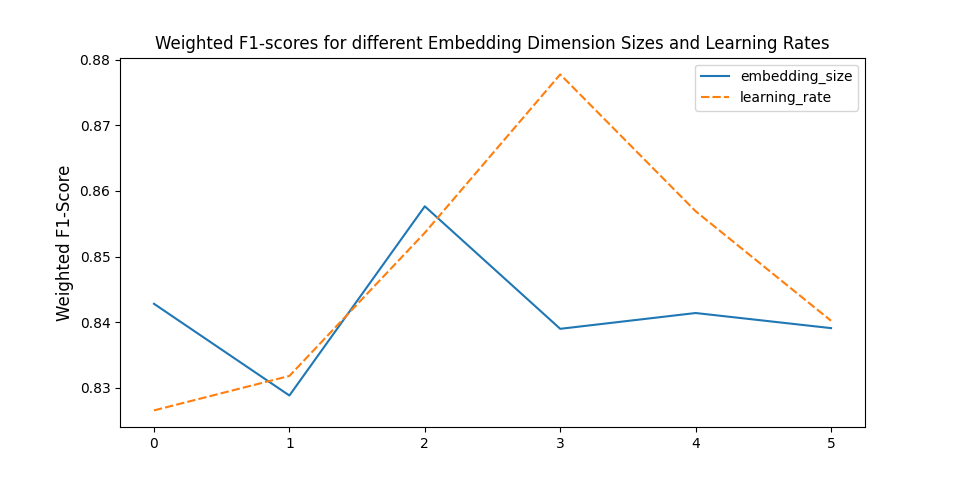
The motivation for building this network came from testing the different embedding dimension sizes. I thought that a model which incorporates multiple dimension sizes might benefit from the addition of different “streams”. The first major change to the network is to connect 4 different LSTMs in parallel between the input and the output layer. The embedding dimension sizes used here are 20, 40, 80, and 160.

A further addition to the network is a residual layer which connects the outputs of the embedding layer with the output layer using fully connected layers and non-linear activation (described in next section). The motivation for residual connections is to counteract vanishing gradients, if they do play a role at all. It is thought that residual connections might smoothen the loss landscape, thus potentially help stabilizing the learning process.

Finally, the fully-connected layer is altered in three ways. First, there is an additional hidden fully-connected layer. Second, a dropout layer is included after the first fully-connected layer. Dropout is thought to help prevent overfitting by resetting connections randomly (with a certain probability). One can think of this feature as adding a stochastic element to the learning process. And last, after the first two fully-connected layers, a non-linearity is introduced: the SiLU function, which is a non-linear, differentiable alternative to (leaky)ReLU, that deals well with vanishing gradients and typically performs as well or better than leakyReLU.

To test this “improved” model, I ran it through the same process as with the standard model, namely testing the model by changing 2 different hyperparameters, whereby either the LSTM hidden dimension size is varied (sizes of 25, 50, 75, 100, 125, and 150 with a constant learning rate of 0.01) or the learning rate is varied (rates of 0.001, 0.003, 0.01, 0.017, 0.03, and 0.1 with a constant hidden dimension size of 100).

The following plot shows the result after testing the new model for 10 epochs, using a batch size of 5.



It appears that changing the hidden dimension size (**blue line, title and legend should indicate hidden dimension size, not embedding dimension size**), doesn’t drastically change the weighted f1-scores, however, there does seem to be an optimum between 50 and 100, where the model can already outperform the models presented in the paper. But most notably, varying the learning rate seems to have a larger impact on the performance of the model (such that it would significantly outperform the other models tested on the WikiNews dataset). What this tells us is that a) this model could benefit much from some tuning or by selecting a different optimizer, and b) that it is to some degree capable of outperforming other canonical ML models (by which I mean, non Neural Network-based models), but also the model included with the assignment and thus, could be considered an “improvement”. There remains nonetheless much which can be tuned, e.g. the optimal dropout probability, batch size, or the correct combination of embedding and hidden dimension sizes within the LSTM subnetworks.

The code is included with the zip file in which this document was found. The relevant pieces of code are ***model/net.py*** and *experiments/base\_model/params.json.*

It is integrated into the same directory structure as used for the group assignment (group 17; the instructions for usign the code can be found in the readme file). Thus, to generate other plots such as above, all of the necessary steps outlined in the readme must be followed.