[Adjusting the implementation 2](#_Toc30772850)

[Batching 2](#_Toc30772851)

[Speedup: 2](#_Toc30772852)

[Experiments and Hyperparameters 3](#_Toc30772853)

[Full overfit on mini-dataset 3](#_Toc30772854)

# 

# Meeting (IA, 24/01/20)

ToDo list:

* complete batching
* lower the learning rate from 0.01 to 10^-3 or -4
* do not plot the loss for each batch, but instead the average training loss over the epoch
* make another overfit test. It should go down all the way to 0
* Include <UNK> in the processing. Initialized as the average of all globals at start
* RRGCN, recurrent RGCN, to add the RNN logic to the Language model task

# Adjusting the implementation

## Batching

In the current version, I do not have real, parallel batching.

I am using a for cycle on the elements of the batch:

**for** i **in** range(len(input\_indices\_lts) - 1):  
 (x, edge\_index, edge\_type) = batch\_rgcn\_input\_ls[i]  
  
 predicted\_globals, predicted\_senses = model(x, edge\_index, edge\_type)

…

Proposal: send batch\_rgcn\_input as the input to the model’s forward().

It depends on the input-to-the-forward-call for each node.

As of now, we collect it in a list for all nodes in the batch as follows:

forward\_input\_ls.append((area\_x, edge\_index, edge\_type))

**Problem**: forward\_input\_ls is a tuple of 3 tensors.

The shapes of (area\_x, edge\_index, edge\_type) are, respectively:

torch.Size([32, 300])  
torch.Size([2, 2048])  
torch.Size([2048])

As they are, they can not be stacked.

They should be padded with -1s, and stacked side-by-side:

torch.Size([32, 300]) 🡪 torch.Size([32, 300])  
torch.Size([2, 2048]) 🡪 torch.Size([32, 2048]) 🡪 torch.Size([32, 4396])  
torch.Size([2048]) 🡪 torch.Size([32, 2048])

Then, each element of the batch can be stacked vertically, thus obtaining a batch-dimension that can be used for parallel processing.

With batch\_size=8, obtain: torch.Size([8, 32, 4396])

### Speedup:

With the sequential version, a batch size of 32 has an iteration time of 0.73/0.90s

# Experiments and Hyperparameters

## Full overfit on mini-dataset