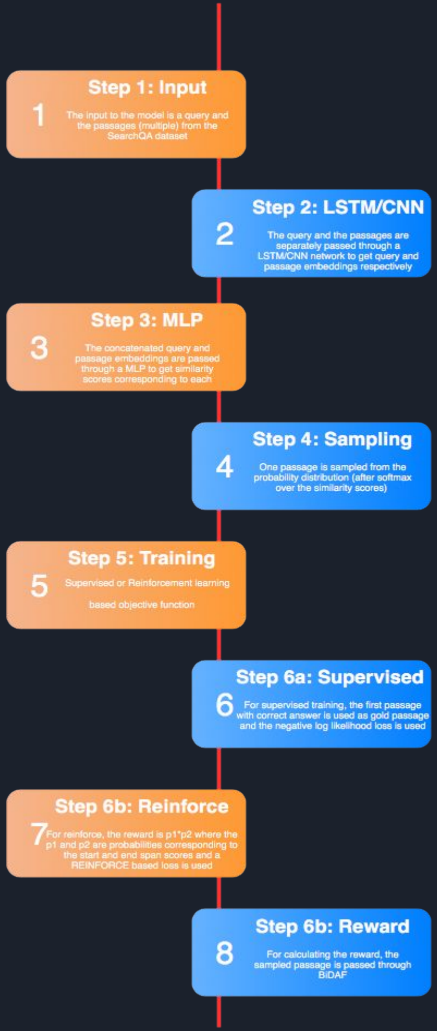
Building end-to-end QA model with reward based objective

TEAM REINFORCE (Prashant and Eti)

Reinforce Pipeline

- Simple model for the reinforcement learning based objective
- Trainable components: LSTM (pre-trained) and MLP
- BiDAF model: pre-trained and kept frozen
- Probability to choose between supervised and RL: (r)^e
    - Where r = 0.8 and e = current epoch - 1



1 **Step 1: Input**
The input to the model is a query and the passages (multiple) from the SearchQA dataset

2 **Step 2: LSTM/CNN**
The query and the passages are separately passed through a LSTM/CNN network to get query and passage embeddings respectively

3 **Step 3: MLP**
The concatenated query and passage embeddings are passed through a MLP to get similarity scores corresponding to each

4 **Step 4: Sampling**
One passage is sampled from the probability distribution (after softmax over the similarity scores)

5 **Step 5: Training**
Supervised or Reinforcement learning based objective function

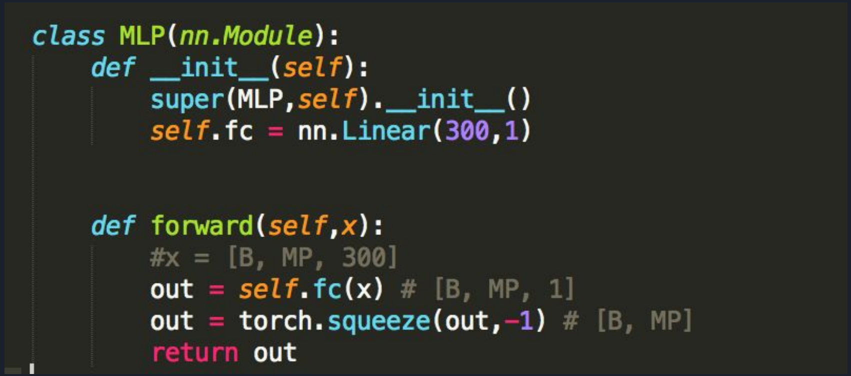
6 **Step 6a: Supervised**
For supervised training, the first passage with correct answer is used as gold passage and the negative log likelihood loss is used

7 **Step 6b: Reinforce**
For reinforce, the reward is p1*p2 where the p1 and p2 are probabilities corresponding to the start and end span scores and a REINFORCE based loss is used

8 **Step 6b: Reward**
For calculating the reward, the sampled passage is passed through BiDAF

Step 1 - 3

- Input to the LSTM: Query and Passages (variable number of passages for each query)
- LSTM output: B x 100 (corresponding to last output state of each query q and each passage p)
- Input to MLP: B x MP x 300 (concatenation of [q, p, q▪p])
- Output of MLP: B x MP



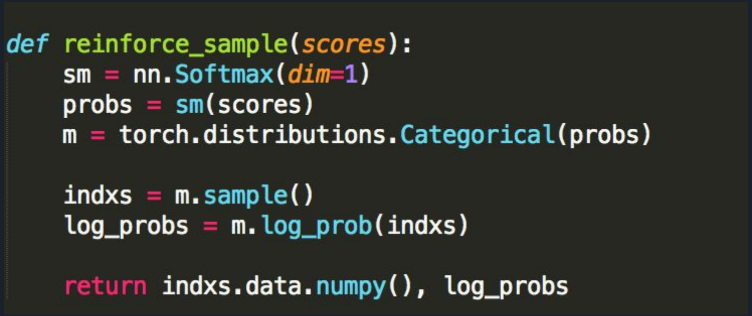
```
class MLP(nn.Module):
    def __init__(self):
        super(MLP,self).__init__()
        self.fc = nn.Linear(300,1)


    def forward(self,x):
        #x = [B, MP, 300]
        out = self.fc(x) # [B, MP, 1]
        out = torch.squeeze(out,-1) # [B, MP]
        return out
```

*MP - maximum number of passages/question in a batch

Step 4 - 5

- For each example in the batch, one passage is sampled based on the similarity scores
- Sampling based on weighted probability distribution (from the similarity scores)
- Similarity scores and Sampling: Independent of the order of passages (and the number of passages)
- Training: Supervised setting (where simple heuristic is used to select gold passage) or RL



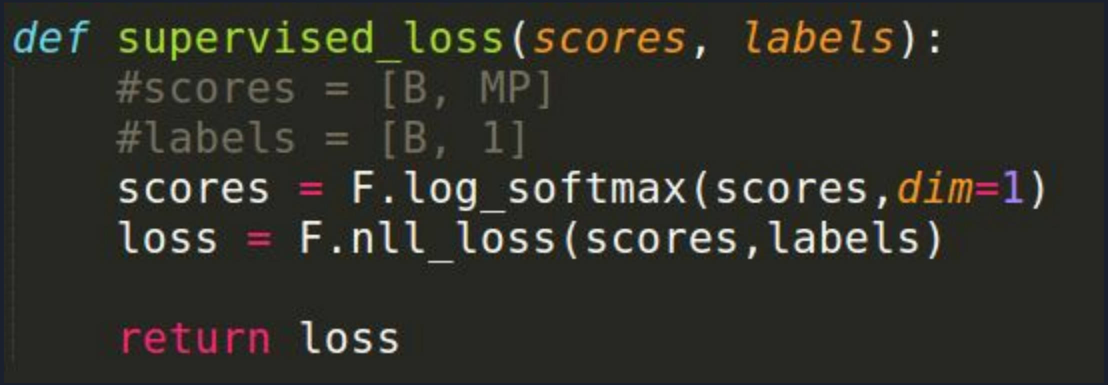
```python
def reinforce_sample(scores):
    sm = nn.Softmax(dim=1)
    probs = sm(scores)
    m = torch.distributions.Categorical(probs)

    indxs = m.sample()
    log_probs = m.log_prob(indxs)

    return indxs.data.numpy(), log_probs
```

Step 6a: Supervised Learning

- For each example in the batch, gold answer is selected as the first passage containing the correct answer
- Each batch is trained using the cross entropy loss.

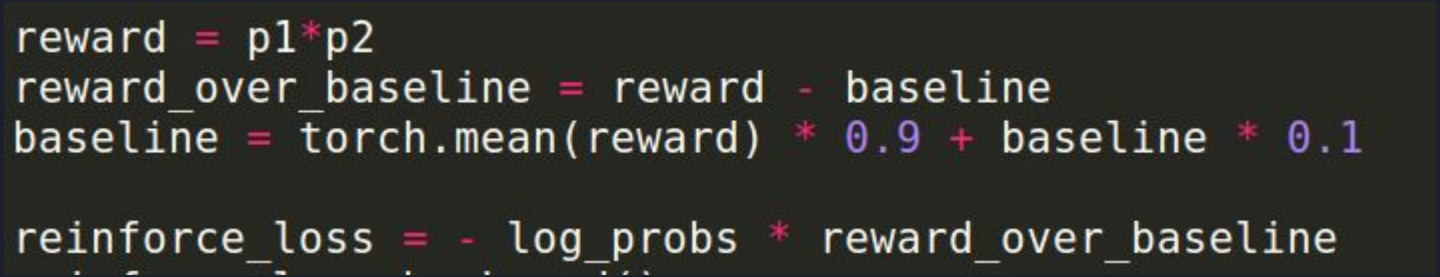


```python
def supervised_loss(scores, labels):
    #scores = [B, MP]
    #labels = [B, 1]
    scores = F.log_softmax(scores,dim=1)
    loss = F.nll_loss(scores,labels)

    return loss
```

Step 6b/c: Reinforcement Learning

- Each passage selected from the passage selection(RL) model is further passed down to the BiDAF model.
- Reward for a passage selection model is the probability across the correct start and end span of answer in the passage
- If the passage does not contains the correct answer, a reward of zero is passed to the model.
- Range of reward will always be between 0 to 1
- REINFORCE loss is implemented by tweaking the cross entropy loss .
- Reward can be considered as a scaling factor used to increase/decrease the log probability of good/bad actions on an average.



```
reward = p1*p2
reward_over_baseline = reward - baseline
baseline = torch.mean(reward) * 0.9 + baseline * 0.1

reinforce_loss = - log_probs * reward_over_baseline
```

Step 6: Training Details

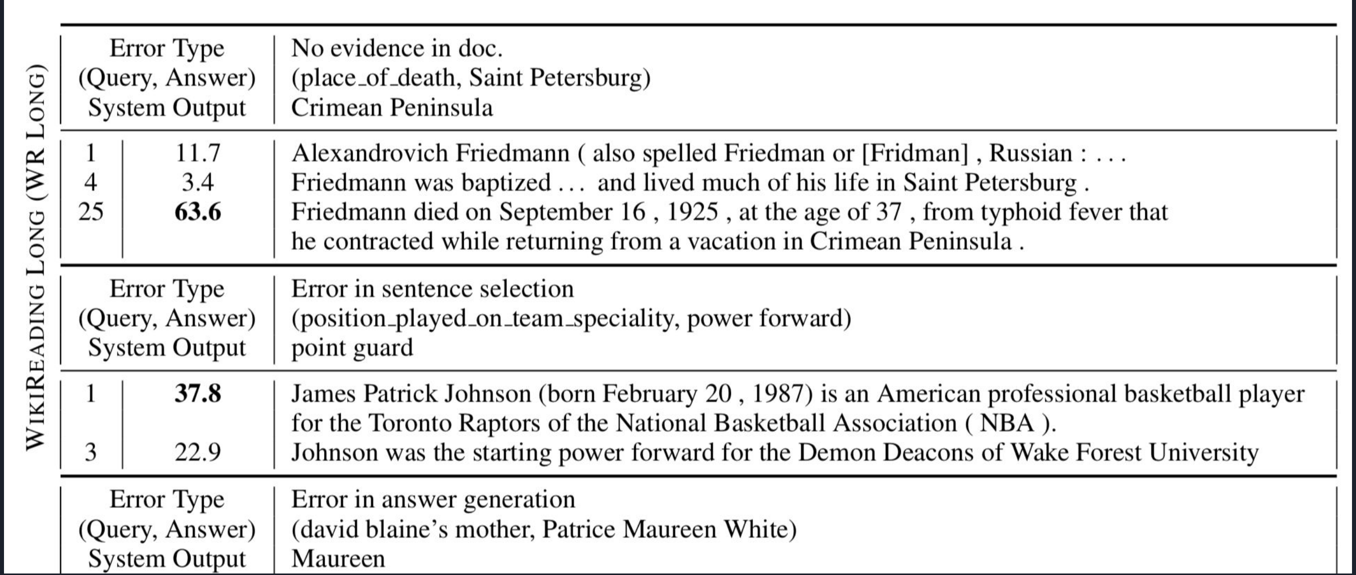
- For each batch example in every epoch: choose between SL or RL based on (r)^e
- Baseline
  - To reduce variance during training and making it stable
  - Heuristic: Running average of reward kept as baseline score
  - Why: Only positive rewards in our case
  - Need to have positive and negative rewards for model to train
  - Actions (passages sampled) getting higher reward (than baseline) -> increase probability of those actions (passages)
  - Modified Reward: r - b (Actions that perform better on an average end up with positive reward)

Tasks in pipeline

1. Write the code for passage ranking(baseline Rl model) **(✓)**
2. Complete and successfully run the code for distant supervision**(✓)**
3. Integrate BiDAF model with RL model **(✓)**
4. Model Tuning **(in progress)**
5. Drafting the final report **(in progress)**

Next week: Agenda

- Show how the probability distribution actually changes during training
- Perform error analysis to figure out the error classes and reasons why the model is poorly performing



WikiReading Long (WR Long)	Error Type		No evidence in doc.
	(Query, Answer)		(place_of_death, Saint Petersburg)
	System Output		Crimean Peninsula
	1	11.7	Alexandrovich Friedmann (also spelled Friedman or [Fridman] , Russian : . . .
	4	3.4	Friedmann was baptized . . . and lived much of his life in Saint Petersburg .
	25	**63.6**	Friedmann died on September 16 , 1925 , at the age of 37 , from typhoid fever that he contracted while returning from a vacation in Crimean Peninsula .
	Error Type		Error in sentence selection
	(Query, Answer)		(position_played_on_team_speciality, power forward)
	System Output		point guard
	1	**37.8**	James Patrick Johnson (born February 20 , 1987) is an American professional basketball player for the Toronto Raptors of the National Basketball Association (NBA).
	3	22.9	Johnson was the starting power forward for the Demon Deacons of Wake Forest University
	Error Type		Error in answer generation
	(Query, Answer)		(david blaine's mother, Patrice Maureen White)
	System Output		Maureen

Timeline

March 03 - March 12	March 13 - March 30	March 31 - April 15	April 16 - April 30
~~Implementation and training of own BiDAF model in dynet~~	~~-Perform experiments on SQuAD dataset~~	- Analyze model on SearchQA Dataset ~~-Implement the passage ranking functionality~~	-Tuning of RL model -Error Analysis and further improvement - Start working on the report