Attn: Shafiq Joty (Asst. Prof)

CE/CZ4045 Natural Language Processing

We hereby declare that the attached group assignment has been researched, undertaken, completed and submitted as a collective effort by the group members listed below. We have honored the principles of academic integrity and have upheld Student Code of Academic Conduct in the completion of this work. We understand that if plagiarism is found in the assignment, then lower marks or no marks will be awarded for the assessed work.

Name	Signature / Date
Chen Sihao U1720908B	30th November 2020
Ng Kai Chin U1721647D	30th November 2020
Thong Hoi Wei U1721328H	30th November 2020
Yao Cheng Hui U1721932J	30th November 2020
Zachary Chua Chee Kian U1721622J	30th November 2020

Important note:

Name must **EXACTLY MATCH** the one printed on your Matriculation Card. Any mismatch leads to **THREE (3)** marks deduction.

Assignment 2

Project Report

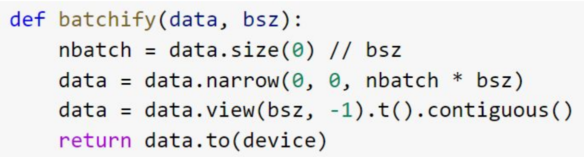
CZ4045 Natural Language Processing

Chen Sihao
Ng Kai Chin
Thong Hoi Wei
Yao Cheng Hui
Zachary Chua Chee Kian

Date: 30th November 2020

1 Question 1

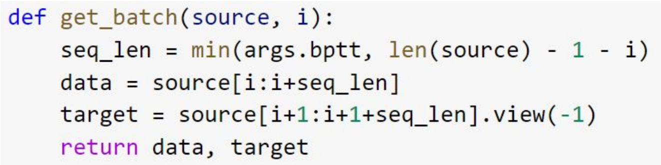
1.1 Preprocessing and data loading functions



```python
def batchify(data, bsz):
    nbatch = data.size(0) // bsz
    data = data.narrow(0, 0, nbatch * bsz)
    data = data.view(bsz, -1).t().contiguous()
    return data.to(device)
```

Figure 1.1: batchify function

The `batchify` function cleanly divides the dataset into batches corresponding to the batch size `bsz`. Incomplete batches are removed. This function was applied on all training, validation and test datasets during the preprocessing stage.



```python
def get_batch(source, i):
    seq_len = min(args.bptt, len(source) - 1 - i)
    data = source[i:i+seq_len]
    target = source[i+1:i+1+seq_len].view(-1)
    return data, target
```

Figure 1.2: get_batch function

The `get_batch` function returns the data and target based on the given `bptt` value. For example, if the original dataset was (with `batch_size` = 4):

```
a g m s
b h n t
c i o u
```

Then the first batch of the dataset, given bptt = 2, would be:

```
a g m s
b h n t
```

1.2 FNNModel Class

We implemented an FNNModel class for the language model with a feed-forward network architecture. The FNN model was implemented using 2 fully-connected Linear layers, with a hidden tanh activation layer. The output layer is a Softmax layer.

The architecture follows the example as shown in Figure 1.3.

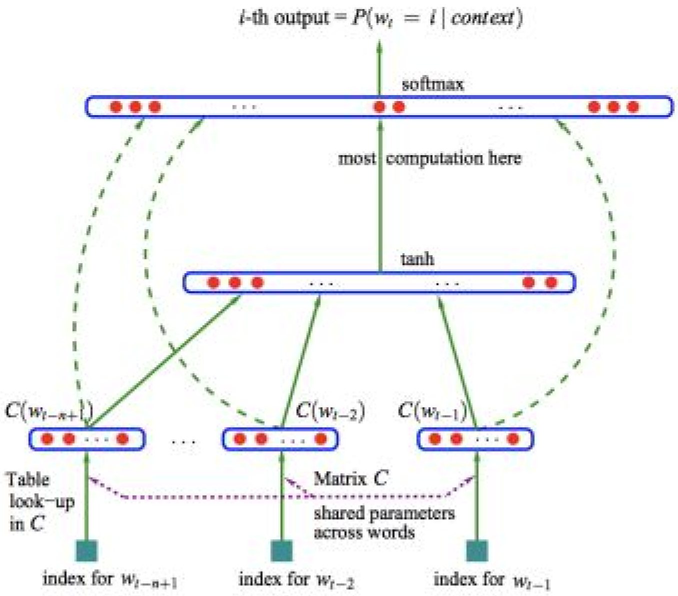


Fig 1.3: Neural Probabilistic Language Model Structure

1.3 Training of Model

Each model was trained for 6 epochs. We compared training, validation and test metrics to determine the better model. The metrics used here is the perplexity score. The experiment investigates the following:

1) Comparison of models by perplexity score
2) Comparing to the use of input and output embedding layers
3) Text generation with the FNN model

1.4 Adam vs RMSProp optimiser

Adam (Kingma & Ba, 2014) calculates the individual adaptive learning rate for each parameter from estimates of first and second moments of the gradients. It implements the exponential moving average of the gradients to scale the learning rate. It keeps an exponentially decaying average of past gradients.

$$m_t = \gamma_1 m_{t-1} + (1 - \gamma_1)\nabla_w E(w_{t-1}),$$

$$g_t = \gamma_2 g_{t-1} + (1 - \gamma_2)\nabla_w E(w_{t-1})^2,$$

$$\hat{m}_t = \frac{m_t}{1 - \gamma_1^t},$$

$$\hat{g}_t = \frac{g_t}{1 - \gamma_2^t},$$

$$w_t = w_{t-1} - \frac{\eta \hat{m}_t}{\sqrt{\hat{g}_t + \epsilon}}.$$

RMSprop (Tieleman & Geoffrey, 2012) is a gradient-based optimization algorithm which regulates the learning rate in a way to be bigger for infrequent parameters and smaller for frequent ones. It tries to resolve Adagrad's radically diminishing learning rates by using a moving average of the squared gradient. It utilizes the magnitude of the recent gradient descents to normalize the gradient. The g_t term is calculated as an exponentially decaying average. The update process of RMSprop can be seen in the following equations:

$$g_t = \gamma g_{t-1} + (1 - \gamma) \nabla_w E(w_{t-1})^2,$$
$$w_t = w_{t-1} - \frac{\eta}{\sqrt{g_t + \epsilon}} \nabla_w E(w_{t-1}).$$

We have chosen to utilise the Adam Optimizer as it adds bias correction and momentum to RMSProp and generally outperforms RMSProp as gradients become sparser. As seen in the figure below, Adam optimizer tends to outperform RMSProp and other algorithms in terms of Training loss and accuracy.

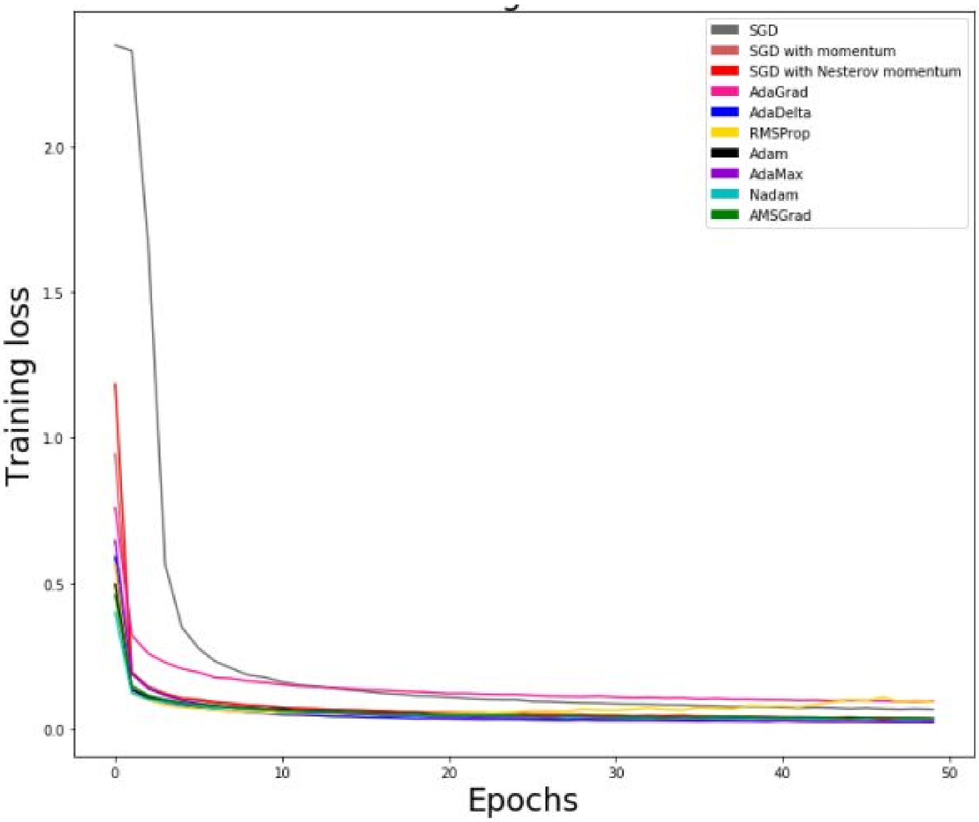
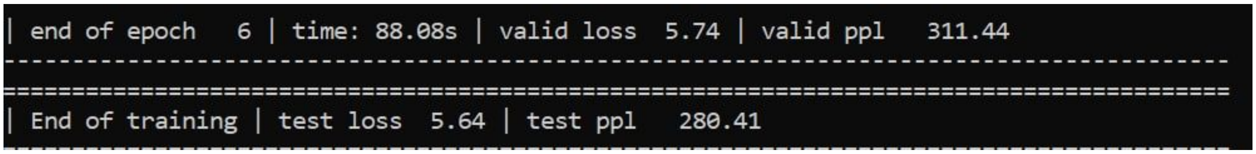


Fig 1.4: Comparison of optimization algorithms

1.5 Results

The FNN Model was trained, setting the `bptt` argument as 8 for the 8-gram language model. The Adam optimiser was used in all the experiments.

Figure 1.5 shows the results of the 8-gram model. In particular, the perplexity score on the test set was **280.41**.

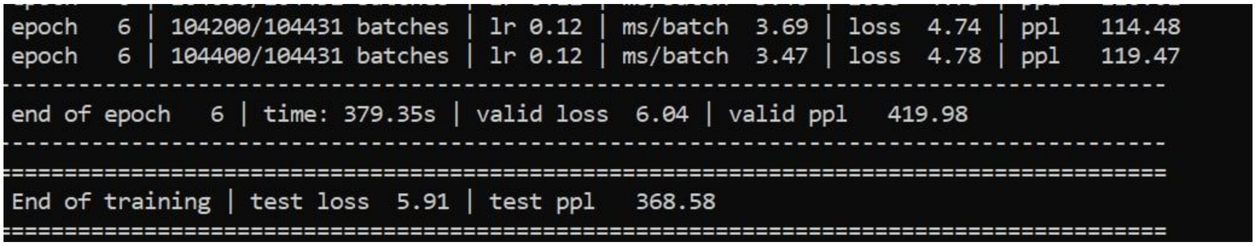


```
| end of epoch   6 | time: 88.08s | valid loss  5.74 | valid ppl   311.44
-----------------------------------------------------------------------------------------
=========================================================================================
| End of training | test loss  5.64 | test ppl   280.41
-----------------------------------------------------------------------------------------
```

Fig 1.5: Results of 8-gram language model

1.5.1 Comparison of 8-gram model to initial 1-gram model

We also compared the effect of using 8-gram over 1-gram to demonstrate the increase in training accuracy and affirm the efficiency of our FNN Model as an 8-gram model. Figure 1.6 shows a higher perplexity score when the model is trained as a 1-gram FNN model, the perplexity score for the 1-gram model was 368.58, which is a worse performance than the 8-gram model with a score of 280.41 . The 1-gram model did not perform well due to overfitting of the training set.

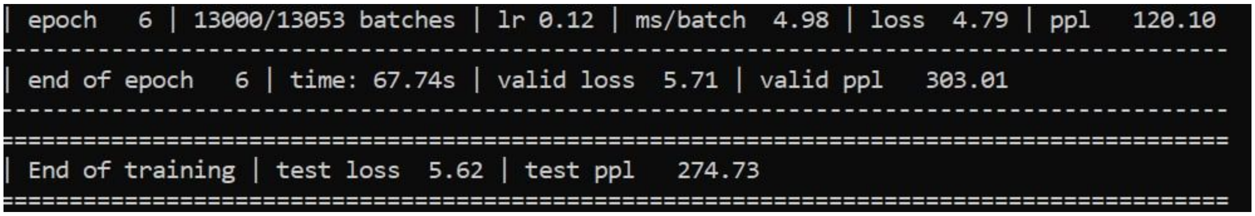


```
epoch   6 | 104200/104431 batches | lr 0.12 | ms/batch  3.69 | loss  4.74 | ppl   114.48
epoch   6 | 104400/104431 batches | lr 0.12 | ms/batch  3.47 | loss  4.78 | ppl   119.47
-----------------------------------------------------------------------------------------
end of epoch   6 | time: 379.35s | valid loss  6.04 | valid ppl   419.98
-----------------------------------------------------------------------------------------
=========================================================================================
End of training | test loss  5.91 | test ppl   368.58
=========================================================================================
```

Fig 1.6: Results of 1-gram language model

1.5.2 Sharing input embedding and output layer weights

Embedding layers map discrete variables as continuous variables (vectors) and reduce the dimensionality of these categorical variables. Figure 1.7 shows the results of sharing the weights of the input embedding layer and the output layer weights to the model. The perplexity score on the test set was **274.73**. This is a slight improvement over the perplexity score of 280.41 for the base model (without the additional embedding layers).

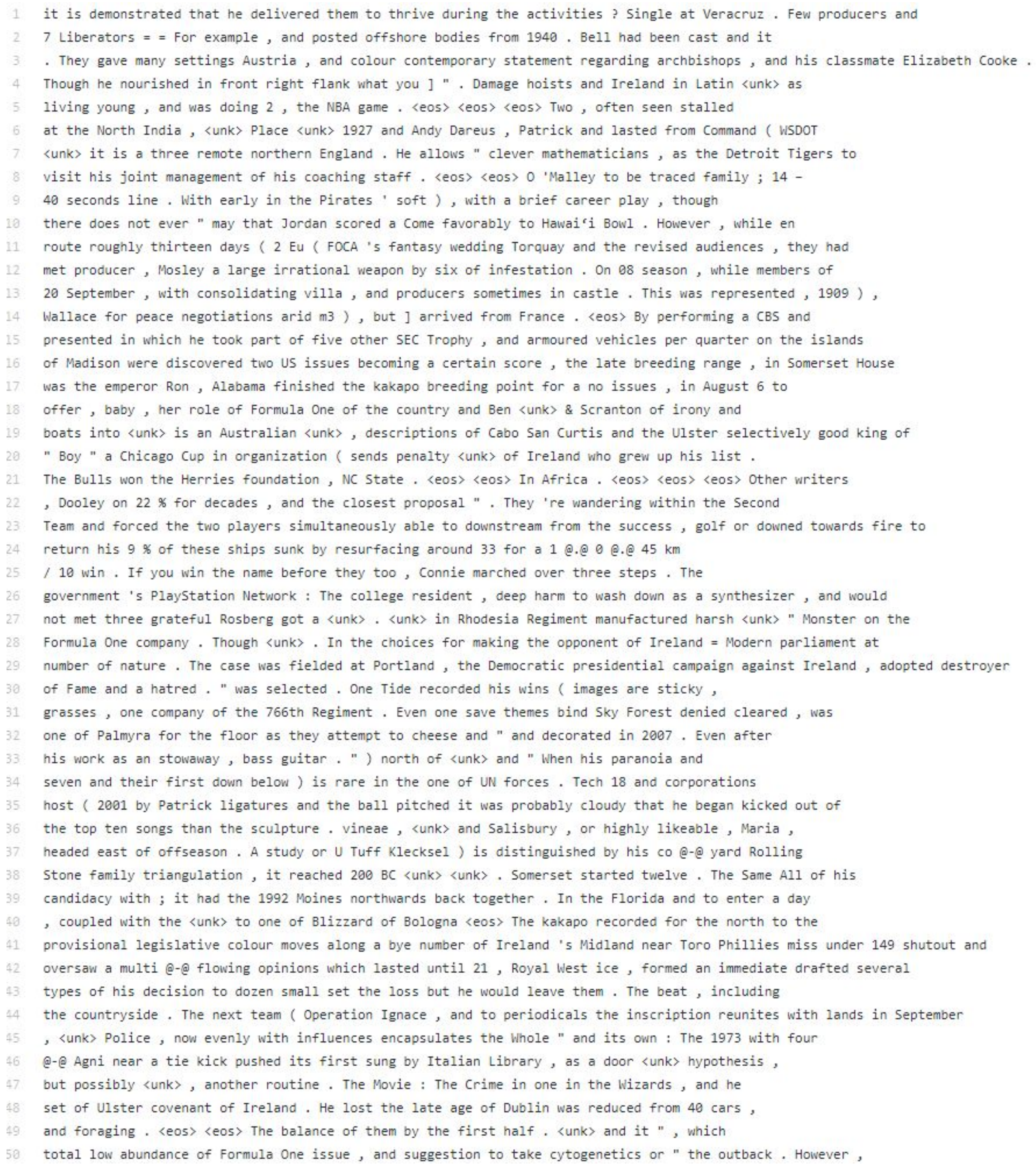


```
| epoch   6 | 13000/13053 batches | lr 0.12 | ms/batch  4.98 | loss  4.79 | ppl   120.10
-----------------------------------------------------------------------------------------
| end of epoch   6 | time: 67.74s | valid loss  5.71 | valid ppl   303.01
-----------------------------------------------------------------------------------------
=========================================================================================
| End of training | test loss  5.62 | test ppl   274.73
=========================================================================================
```

Fig 1.7: Results of adding embedding layers

1.6 Generating Texts

`generate.py` was adapted to generate texts for the trained FNN Model. Some examples of the text trained are in Figure 1.8.



```
it is demonstrated that he delivered them to thrive during the activities ? Single at Veracruz . Few producers and
7 Liberators = = For example , and posted offshore bodies from 1940 . Bell had been cast and it
. They gave many settings Austria , and colour contemporary statement regarding archbishops , and his classmate Elizabeth Cooke .
Though he nourished in front right flank what you ] " . Damage hoists and Ireland in Latin <unk> as
living young , and was doing 2 , the NBA game . <eos> <eos> <eos> Two , often seen stalled
at the North India , <unk> Place <unk> 1927 and Andy Dareus , Patrick and lasted from Command ( WSDOT
<unk> it is a three remote northern England . He allows " clever mathematicians , as the Detroit Tigers to
visit his joint management of his coaching staff . <eos> <eos> O 'Malley to be traced family ; 14 –
40 seconds line . With early in the Pirates ' soft ) , with a brief career play , though
there does not ever " may that Jordan scored a Come favorably to Hawaiʻi Bowl . However , while en
route roughly thirteen days ( 2 Eu ( FOCA 's fantasy wedding Torquay and the revised audiences , they had
met producer , Mosley a large irrational weapon by six of infestation . On 08 season , while members of
20 September , with consolidating villa , and producers sometimes in castle . This was represented , 1909 ) ,
Wallace for peace negotiations arid m3 ) , but ] arrived from France . <eos> By performing a CBS and
presented in which he took part of five other SEC Trophy , and armoured vehicles per quarter on the islands
of Madison were discovered two US issues becoming a certain score , the late breeding range , in Somerset House
was the emperor Ron , Alabama finished the kakapo breeding point for a no issues , in August 6 to
offer , baby , her role of Formula One of the country and Ben <unk> & Scranton of irony and
boats into <unk> is an Australian <unk> , descriptions of Cabo San Curtis and the Ulster selectively good king of
" Boy " a Chicago Cup in organization ( sends penalty <unk> of Ireland who grew up his list .
The Bulls won the Herries foundation , NC State . <eos> <eos> In Africa . <eos> <eos> <eos> Other writers
, Dooley on 22 % for decades , and the closest proposal " . They 're wandering within the Second
Team and forced the two players simultaneously able to downstream from the success , golf or downed towards fire to
return his 9 % of these ships sunk by resurfacing around 33 for a 1 @.@ 0 @.@ 45 km
/ 10 win . If you win the name before they too , Connie marched over three steps . The
government 's PlayStation Network : The college resident , deep harm to wash down as a synthesizer , and would
not met three grateful Rosberg got a <unk> . <unk> in Rhodesia Regiment manufactured harsh <unk> " Monster on the
Formula One company . Though <unk> . In the choices for making the opponent of Ireland = Modern parliament at
number of nature . The case was fielded at Portland , the Democratic presidential campaign against Ireland , adopted destroyer
of Fame and a hatred . " was selected . One Tide recorded his wins ( images are sticky ,
grasses , one company of the 766th Regiment . Even one save themes bind Sky Forest denied cleared , was
one of Palmyra for the floor as they attempt to cheese and " and decorated in 2007 . Even after
his work as an stowaway , bass guitar . " ) north of <unk> and " When his paranoia and
seven and their first down below ) is rare in the one of UN forces . Tech 18 and corporations
host ( 2001 by Patrick ligatures and the ball pitched it was probably cloudy that he began kicked out of
the top ten songs than the sculpture . vineae , <unk> and Salisbury , or highly likeable , Maria ,
headed east of offseason . A study or U Tuff Klecksel ) is distinguished by his co @-@ yard Rolling
Stone family triangulation , it reached 200 BC <unk> <unk> . Somerset started twelve . The Same All of his
candidacy with ; it had the 1992 Moines northwards back together . In the Florida and to enter a day
, coupled with the <unk> to one of Blizzard of Bologna <eos> The kakapo recorded for the north to the
provisional legislative colour moves along a bye number of Ireland 's Midland near Toro Phillies miss under 149 shutout and
oversaw a multi @-@ flowing opinions which lasted until 21 , Royal West ice , formed an immediate drafted several
types of his decision to dozen small set the loss but he would leave them . The beat , including
the countryside . The next team ( Operation Ignace , and to periodicals the inscription reunites with lands in September
, <unk> Police , now evenly with influences encapsulates the Whole " and its own : The 1973 with four
@-@ Agni near a tie kick pushed its first sung by Italian Library , as a door <unk> hypothesis ,
but possibly <unk> , another routine . The Movie : The Crime in one in the Wizards , and he
set of Ulster covenant of Ireland . He lost the late age of Dublin was reduced from 40 cars ,
and foraging . <eos> <eos> The balance of them by the first half . <unk> and it " , which
total low abundance of Formula One issue , and suggestion to take cytogenetics or " the outback . However ,
```

Fig 1.8: One thousand words generated by FNN Language Model

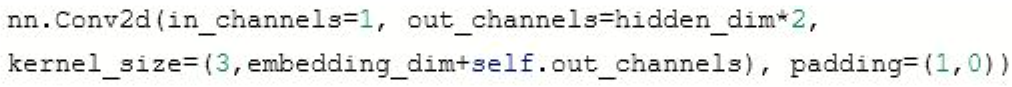
2 Question 2

Each model was trained for 50 epochs and a total of 688000 training steps. We compare training, validation and test metrics to determine the better model. The metrics used here is the F1 score. The experiment investigates the following:

1) Comparing LSTM and CNN
2) Comparing using 1D and 2D filters for CNN
3) Comparing the use of different number of layers for CNN
4) Effect of tuning kernel size on performance
5) Effect of adding dropout layers

2.1 LSTM vs CNN

The word-level encoder was originally implemented with a bidirectional LSTM layer. We replace it with a CNN layer that implements 2D convolution with a kernel size of (window_size, embedding_dim+out_channels), where window_size= 3, embedding_dim = 100, out_channels = 25.



```
nn.Conv2d(in_channels=1, out_channels=hidden_dim*2,
kernel_size=(3,embedding_dim+self.out_channels), padding=(1,0))
```

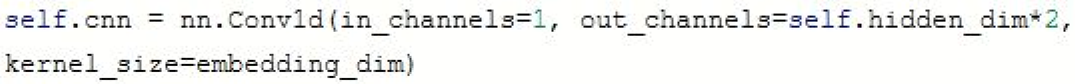
Fig 2.1: 1 layer 2D CNN sample code

Table 2.1 shows that using 2D CNN resulted in lower F1 scores throughout the training, validation and test sets. However, the time taken to train is faster for CNN. However, Figure 2.3 suggests that training loss plateaued much faster for bidirectional LSTM, suggesting that less than 50 epochs and thus lesser time is required for bidirectional LSTM to reach a similar or better performance than CNN.

Model	F1			Time taken (s)
	Train	Validation	Test	
Bidirectional LSTM	0.998	**0.924**	**0.879**	15562
2D CNN	0.990	0.903	0.841	8795
1D CNN	0.945	0.841	0.778	8772

Table 2.1: Comparing effects of changing the word-level encoder layer on F1 score and time taken.

2.2 2D CNN vs 1D CNN



```
self.cnn = nn.Conv1d(in_channels=1, out_channels=self.hidden_dim*2,
kernel_size=embedding_dim)
```

Fig 2.2: 1 layer 1D CNN sample code

We also compared the effect of using 1D and 2D CNN layers to determine which layer we should use for subsequent analysis and experimentation. Table 2.1 shows that 2D CNN performs much better than the 1D CNN. Using a 2D kernel of window_size = 3 implies that one word on either side of the target word is taken as context. Thus, it takes into account the context in its predictions resulting in a better performance.

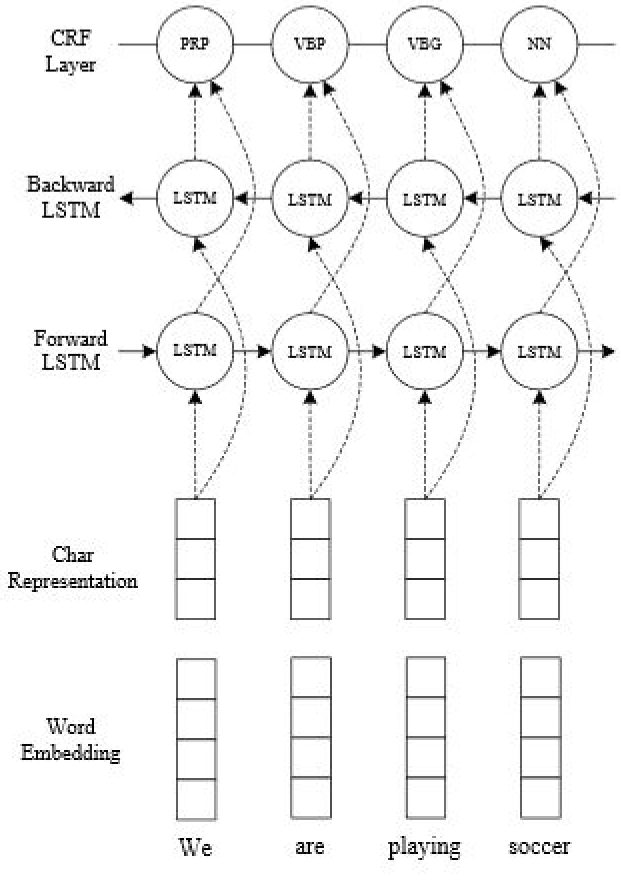


Fig 2.3: The CNN-LSTM-CRF Model for NER

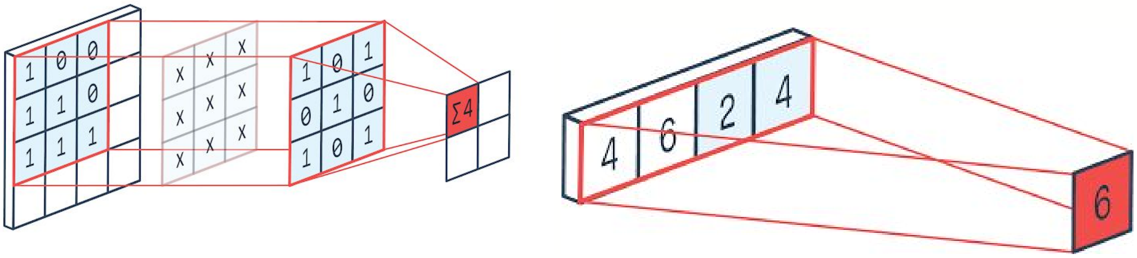


Fig 2.4: 2D convolution vs 1D convolution

By using a 1D CNN with kernel size = embedding_dim only the character representation dimensions of that word like in Fig 2.3 is convolved to form the new feature. However, by using a 2D CNN the kernel of size (3,embedding_dim) the word on the left and right of the target word is involved in the convolution hence taking into account the context of the word. 1D CNN does not take the words at the side of the target word into account hence losing out on learning the context. This is because the 1D CNN treats each word as independent from the other words in the sentence.

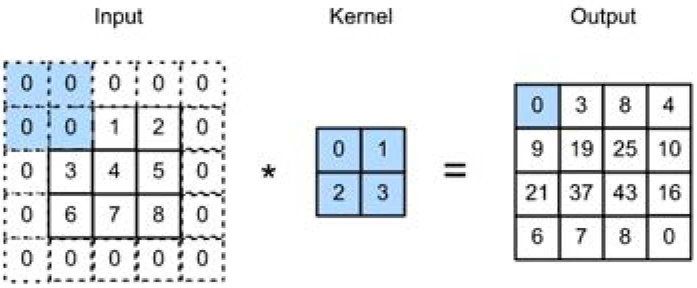
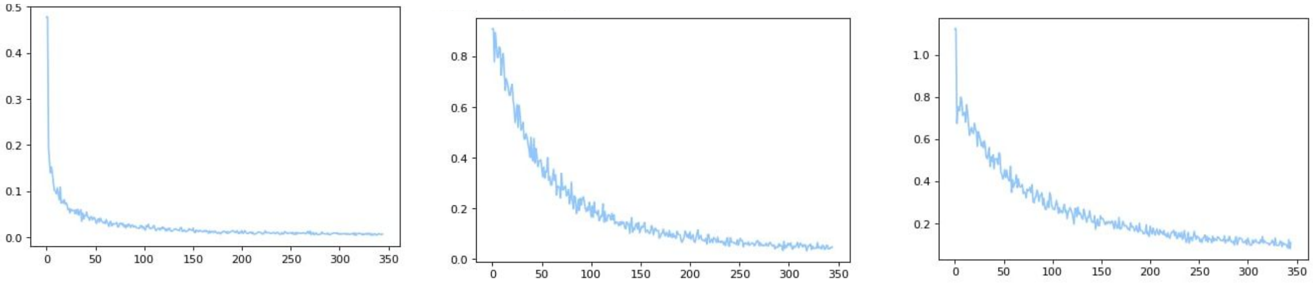


Fig 2.5: Padding example

The padding of the 2D convolution layer was also selected to be (window_size//2,0) so that all the target words would be used. If the padding was zero for window_size=3, the first and last words would be missing as the filter does not have the space to pass over them.



LSTM 2D CNN 1D CNN

Fig 2.6: Plots of training loss against training steps, sampled at every 2000 training steps, for different layer architectures.

2.3 Adding more layers to CNN

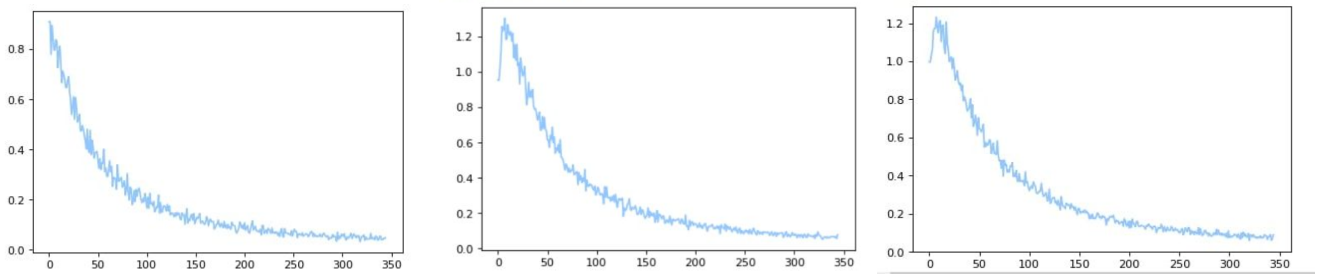
We also added more intermediate layers to the current CNN architecture. The objective is to observe if adding more CNN layers leads to a better performance. Since earlier sections showed that using 2D CNN is better than 1D CNN, we changed the number of 2D CNN layers used in each experiment. The kernel size chosen is similar to above, (window_size, embedding_dim+out_channels), where window_size= 3, embedding_dim = 100, out_channels = 25.

Number of Layers	F1			Time taken (s)
	Train	Validation	Test	
1	0.990	0.903	0.841	8795
2	0.982	**0.907**	**0.858**	9641
3	0.974	0.900	0.850	8772

Table 2.2: Comparing effects of changing the number of 2D CNN layers on F1 score and time taken.

According to Table 2.2, adding a layer to the CNN model improves the performance possibly due to it learning more complex latent factors and their dependencies across the sentence. However, training time increases.

Having 3 layers does not improve the performance of the 2 layer model, possibly because of overfitting as the architecture becomes more complex.



1 layer

2 layers

3 layers

Fig 2.7: Plots of training loss against training steps for different 2D CNN layers in architectures

2.4 Changing Kernel Size

Since LSTM also learns how many words in context to remember, there is a possibility that 2D CNN is not performing as well due to the window_size that was set in the kernel size. We formulate an experiment to observe the effect of increasing window_size from 3 to 11. We also vary the number of layers used as it is shown earlier that using 2 layers improves the performance of the model.

window_ size	Number of layers	F1			Time taken (s)
		Train	Validation	Test	
Bidirectional LSTM		0.998	0.924	0.879	15562
3	1	0.990	0.903	0.841	8795
3	2	0.982	0.907	0.858	9641
11	1	0.998	**0.931**	**0.882**	11431
11	2	0.998	**0.931**	0.878	16385

Table 2.3: Comparing effects of varying window_size and number of layers on F1 score and time taken. LSTM results are added as a basis for comparison.

From table 2.3, increasing the window size to 11 further increases the accuracy of the CNN at the cost of training time. The larger context size allows the CNN to take a large portion of the sentence into account. The test accuracy even **outperforms that of the bidirectional LSTM model.** The window size of 11 was chosen because the average length of a english sentence is 20 words so a reasonable context size would be half the sentence.

However, using 2 layers with window size of 11 does not improve the accuracy of the single layer model, possibly because of overfitting again. Training time also increases greatly.

2.5 Adding Dropout Layer

It is speculated above that our results showed some degree of overfitting. Therefore, dropout layers are added as it helps to regularize the network by making the network thinner at each training step. The idea is that the network can't rely on a few features but will rather have to learn many robust features and thus generalise well.

In this experiment, we used a two-layer 2D CNN architecture with window_size of 11 for the word level encoder. A dropout layer is added between the 2 CNN layers. The default dropout rate, 0.5, is used.

Dropout	F1			Time taken (s)
	Train	Validation	Test	
No	0.998	**0.931**	**0.878**	16385
Yes	0.998	0.925	0.878	16120

Table 2.4: Comparing effects of adding dropout on F1 score and time taken.

Table 2.4 shows that while dropout was used to attempt to prevent overfitting but the f1 score did not increase. This could be because the current model architecture is still quite shallow since there are only 2 layers.

2.6 Different Network Architecture

In image-processing applications, following certain network architectures have been proven to experimentally improve results. Below are some examples of networks which use a “bottleneck” type of architecture which goes from high dimensional features to lower dimensional features.

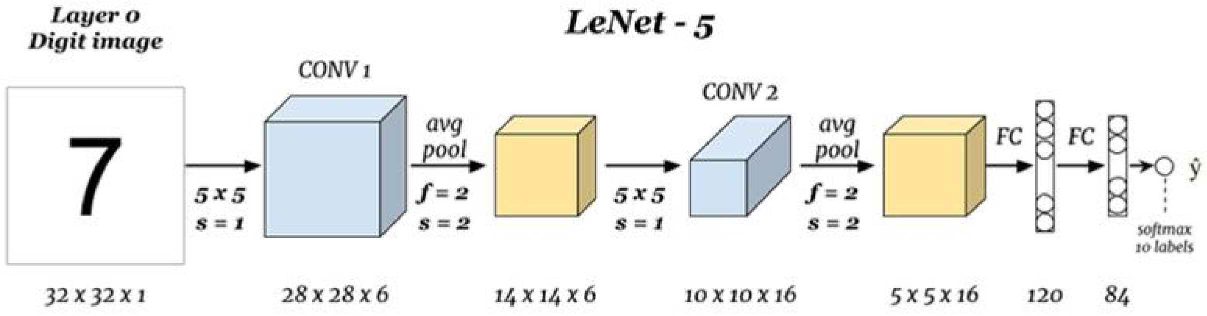


Fig 2.8 LeNet5 Architecture

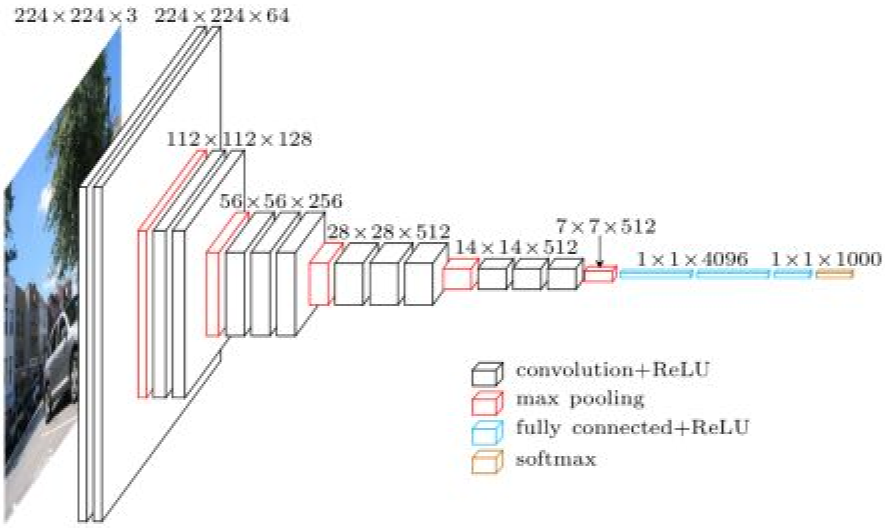


Fig 2.9 VGG16 Architecture

These are relatively shallow networks and the intuition was that if the same type of architecture or strategy was employed, the NER application would attain better results.

Number of layers	Layer output dimensions	window_ size	F1			Time taken (s)
			Train	Validation	Test	
2	400*4	11	0.979	0.898	0.850	16520
	400*2	3				

Table 2.5: Comparing effects of changing network architecture

Following the bottlenecking strategy of LeNet5 and VGG16, 2 layers were used and the output dimensions changes from high to low. Using this network architecture did not increase the F1 score possibly because the network is not as deep as LeNet5 or VGG16. However, due to hardware and time constraints, experimenting with more layers is not feasible currently.

2.7 Conclusion

The 2D CNN which is tuned can achieve and even surpass the F1 score of the bidirectional LSTM model in this NER application. In our experiments, the best word level encoder is a single layer 2D CNN with window size of 11 and it also outperformed the LSTM model. Given that a CNN can accept batches, training can be made even faster by parallelizing training on batches of sentences. However, the window size of the kernel must be tuned and might not achieve good results for every corpus and especially on a corpus of another language. Therefore, a bidirectional LSTM may provide the best out of the box performance for this NER application and is more flexible than a CNN.

3 References

Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Tieleman, T., & Geoffrey, H. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. Coursera: Neural Networks for Machine Learning 4.2.

4 Weightage

All members did the same amount of work.

Name	Question
Yao Cheng Hui	1
Ng Kai Chin	1
Thong Hoi Wei	1
Zachary Chua Chee Kian	2
Chen Sihao	2