Notebook contents

It may be challenging to look at the notebook contents without a streamlined content index or illustration. So we thought it might be a good idea to create what each notebook does in detail. Readers interested in Notebook codes are encouraged to refer to this and they will be able to follow every point raised in the paper. This notebook also contains some sections not presented in the paper such as the binary training case that we thought of including in the paper, but didn’t. Interested readers can also take a look at those findings..

We have 3 independent notebook files.

**A.nlp\_project\_final.ipynb**

**B.decomposable\_attention.ipynb**

**C.visualizations.ipynb**

Due to the large page volumes, **A.nlp\_project\_final.ipynb** will be particularly hard to follow without this file, so readers are encouraged to refer to this. If you read this, **B.decomposable\_attention.ipynb** and **C.visualizations.ipynb** will be easy to follow.

1. The purpose of 1 is to verify the pretrained model’s ability on our data set. Because the model seemed to do a really good job before and after the training on MNLI+SNLI, the need to test the pre-trained model’s basic ability became apparent. Even without any training the model can do a good job on Glockner’s dataset with evaluation score of 0.71

2-1. The purpose of 2-1 is to have the base model(3 labels) trained with only MNLI. This base model will be saved in "models/google/electra-small-discriminator" in our shared folder. More evaluations are done in 2-2(a) when we load the model trained here

2-2(a). The purpose of 2-2(a) is to load the base model(MNLI) we trained in 2-1, and this way we don’t have to train and can just load the model from our shared folder "models/google/electra-small-discriminator"

The base model(MNLI) evaluation accuracy is,

MNLI eval\_accuracy: 0.83

SNLI eval\_accuracy: 0.77

Glockner eval\_accuracy: 0.90

Interestingly, the model does a decent job on SNLI even if it was never trained on it, meaning it could lead to a transfer of “similarly defined task”, which becomes very important in later part of the notebook when we see there is almost no transfer of ”not similarly defined task “

Also, Glockner’s dataset evaluation accuracy is at 0.90 from the pretrained model’s performance of 0.71

2-2(b). The purpose of 2-2(b) is to evaluate the performance of the base model(MNLI) on HANSdateset. This code will save the performance text file 'hans\_predictions\_base.txt'​​ .

The result below,

The base model(MNLI) Heuristic score as below:

Heuristic entailed results:

lexical\_overlap: 0.8238

subsequence: 0.9716

constituent: 0.9776

Heuristic non-entailed results:

lexical\_overlap: 0.4884

subsequence: 0.162

constituent: 0.16

So, It seems the base model(MNLI) does poorly on subsequence & constituent and a little better on lexical\_overlap.

2-3. The purpose of 2-3 is to have the base model(3 labels) trained with MNLI+SNLI. This base model(MNLI+SNLI) will be saved in 'models/mnli\_snli\_base\_trained\_model' in our shared folder. More evaluations are done in 2-3(a) when we load the model trained here

2-4(a). The purpose of 2-4(a) is to load the base model(3 labels, MNLI+SNLI) we trained in 2-3, from 'models/mnli\_snli\_base\_trained\_model'

The base model(3 labels, MNLI+SNLI) evaluation accuracy is,

MNLI eval\_accuracy: 0.83

SNLI eval\_accuracy: 0.89

Glockner eval\_accuracy: 0.95

As expected, the model does a much better job on SNLI from 0.77 to 0.89. Interestingly, it didn’t improve on MNLI. It seems adding SNLI data does not add so much information on the MNLI task.

Also, Glockner’s dataset evaluation accuracy rose from 0.90 to 0.95, so there was something to gain from the data.

2-4(b). The purpose of 2-4(b) is to evaluate the performance of the base model(3 labels, MNLI+SNLI) on HANSdateset. This code will save the performance text file 'hans\_snli+mnli\_base.txt'​​ .

The result below,

The base model(3 labels, MNLI+SNLI) Heuristic score as below:

Heuristic entailed results:

lexical\_overlap: 0.9198

subsequence: 0.9966

constituent: 0.9894

Heuristic non-entailed results:

lexical\_overlap: 0.3412

subsequence: 0.0362

constituent: 0.061

Interestingly, The base model(3 labels, MNLI+SNLI) does worse than The base model(MNLI) for Heuristic non-entailed results. Meaning adding more data actually got a worse result!

3-1. The purpose of 3-1 is to train the model trained with only 2 labels.(entailment or non-entailment) using MNLI. This model will be saved in 'models/just\_heur\_trained\_model' in our shared folder. More evaluations are done in 3-2(a) when we load the model trained here

3-2(a). The purpose of 3-2(a) is to load the model(2 labels, MNLI) we trained in 3-1, from 'models/just\_heur\_trained\_model'

the model(2 labels, MNLI) evaluation accuracy is,

MNLI eval\_accuracy: 0.894

SNLI eval\_accuracy: 0.880

Glockner eval\_accuracy: 0.956

As you might imagine, predicting 2 is easier than predicting 3, so it does lead to a higher score.

3-2(b). The purpose of 3-2(b) is to evaluate the performance of the model(2 labels, MNLI) on HANSdateset. This code will save the performance text file 'hans\_predictions\_mnli\_binary.txt'​​ .

The result below,

The model(2 labels, MNLI) Heuristic score as below:

Heuristic entailed results:

lexical\_overlap: 0.9866

subsequence: 1.0

constituent: 0.9996

Heuristic non-entailed results:

lexical\_overlap: 0.003

subsequence: 0.0042

constituent: 0.0562

Interestingly, training the model with binary labels leads to a worse result for Heuristic non-entailed predictions. It doesn’t seem to be able to do any correct predictions on Heuristic non-entailed results. The base model(3 labels, MNLI+SNLI) at least did some correct prediction on at least lexical overlap, but the model(2 labels, MNLI) is blindly predicting entailment. Later, the model (2 labels, MNLI+SNLI) shows almost the same trend in 4-2(b)

3-3. The purpose of 3-3 is to train the model trained with only 2 labels.(entailment or non-entailment) using MNLI+Hans dataset. This model will be saved in 'models/mnli\_heur\_trained\_model' in our shared folder. More evaluations are done in 3-4(a) when we load the model trained here

3-4(a). The purpose of 3-4(a) is to load the model(2 labels, MNLI+Hans) we trained in 3-3, from 'models/mnli\_heur\_trained\_model’

the model(2 labels, MNLI+Hans) evaluation accuracy is,

MNLI eval\_accuracy: 0.895

SNLI eval\_accuracy: 0.870

Glockner eval\_accuracy: 0.952

The result is almost the same as the model(2 labels, MNLI)

3-4(b). The purpose of 3-4(b) is to evaluate the performance of the model(2 labels, MNLI+Hans) on HANSdateset. This code will save the performance text file 'hans\_predictions\_mnli+hans\_binary.txt'​​ .

The result below,

Heuristic entailed results:

lexical\_overlap: 1.0

subsequence: 1.0

constituent: 1.0

Heuristic non-entailed results:

lexical\_overlap: 1.0

subsequence: 1.0

constituent: 1.0

It does correctly predict all data, but does it mean it learned how Heuristic entailment and Heuristic non-entailment work as intended?

4-1 is to train the model with mnli+snli, but it is binary label training so basically SNLI is added from 3-1

4-2(a). To load the model(2 labels, MNLI+SNLI) from 'models/mnli\_snli\_trained\_model'

the model(2 labels, MNLI+SNLI) evaluation accuracy is,

MNLI eval\_accuracy: 0.895

SNLI eval\_accuracy: 0.933

Glockner eval\_accuracy: 0.981

Similar score for MNLI as in 3.2(a), but SNLI accuracy is better with obvious reason, and also Glockner accuracy is better from 0.95 to 0.98.

This Glockner result shows a similar pattern as with 2-4(a), when we added SNLI, Glockner scores got higher, and it is the same here. So, adding SNLI data must have helped in Glockner’s lexical knowledge.

4-2(b). The purpose of 4-2(b) is to evaluate the performance of the model(2 labels, MNLI+SNLI) on HANSdateset. This code will save the performance text file 'hans\_predictions\_mnli+snli.txt'​​ .

The result below,

The model(2 labels, MNLI+SNLI) Heuristic score as below:

Heuristic entailed results:

lexical\_overlap: 0.9792

subsequence: 1.0

constituent: 0.9912

Heuristic non-entailed results:

lexical\_overlap: 0.0774

subsequence: 0.0204

constituent: 0.0678

This is almost the same result as 3-2(b), adding SNLI didn’t really help in Heuristic non-entailed results.

4.3 is training with Hans from 4.2 data, and 4.4(a) and 4.4(b) shows a similar result as before and nothing of significance is to be noted as it is very similar.