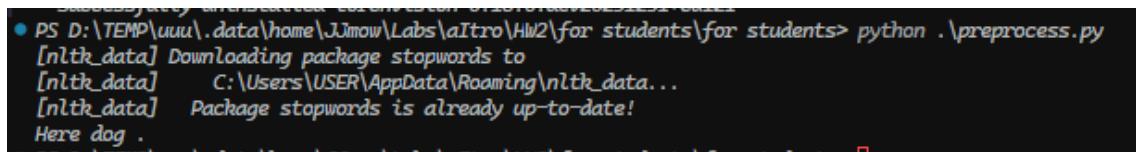


hw2_109700046

- For Part 0
 - Briefly explain the method you implemented and give an example (such as the E.g in the remove_stopwords function) in the report.
 - First, I keep only a-z, A-Z and numbers or . In the text. And than I keep the first word in tokens because I thought the first word is somehow meaningful. Next, I use stem function to do regularization (ran → run, running → run). And than, I remove stop words as usual.
 - Here is a dog.



```
PS D:\TEMP\uiui\data\home\Jimow\Labs\AItra\HW2\for students\for students> python .\preprocess.py
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\USER\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
Here dog .
```

- For Part 1
 - Briefly explain the concept of perplexity in report and discuss how it will be influenced.'
 - perplexity is the indicator of the performance of an NLP model. It measures how good the probability distribution of a model predicting a word sequence, or a sentence. the less confuse the model is, the less perplexity it gets. The model itself can influence perplexity, a good model can make perplexity small, and the size of word sequence during training can also affect perplexity since the larger it is the more probability we will get bigger.
 - Screenshot the outputs and tell your observations about the differences in the perplexity caused by the preprocessing methods (1. without preprocess 2. with remove stop words 3. with your method).

```
PS D:\TEMP\uuu\data\home\JJmow\Labs\AIIntro\HW2\for students\for students> python main.py --model_type ngram --preprocess 0 --part 2
[nltk_data] Downloading package stopwords to
[nltk_data]   C:\Users\USER\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
cuda:0 2.2+cu121
perplexity of ngram: 116.26046015880357
F1 score: 0.7057, Precision: 0.7088, Recall: 0.7065
```

without preprocess

```
PS D:\TEMP\uuu\data\home\JJmow\Labs\AIIntro\HW2\for students\for students> python main.py --model_type ngram --preprocess 1 --part 2
[nltk_data] Downloading package stopwords to
[nltk_data]   C:\Users\USER\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Perplexity of ngram: 195.43245350997685
F1 score: 0.6769, Precision: 0.6926, Recall: 0.6815
```

with remove stop words

```
PS D:\TEMP\uuu\data\home\JJmow\Labs\AIIntro\HW2\for students\for students> python main.py --model_type ngram --preprocess 1 --part 2
[nltk_data] Downloading package stopwords to
[nltk_data]   C:\Users\USER\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Perplexity of ngram: 205.50875584567925
F1 score: 0.6904, Precision: 0.7078, Recall: 0.6951
```

with my method

- the best result is from which wasn't preprocessed. And the perplexity of my preprocess is higher than the default one, though I have better f1-score than remove stop words, both higher than the one not being preprocessed, which means the model is more confused when handling preprocessed data in my case.
- For Part 2:
 - Briefly explain the two pre-training steps in BERT. Self-supervised Learning by Hung-yi Lee tutorial
 - First step → masking, which is switching some words in the input into a special token called 'mask' or random words, then we want to minimum the loss between output of those switched words and the word before being switched during training.
 - Second step → next sentence prediction, when given two adjacent sentences, we want BERT to define which sentence is the former one and which is the latter one, also called SOP, this problem is hard enough for the model to learn some features and become way stronger.
 - Briefly explain four different BERT application scenarios

- sentiment analysis → homework of this time, which is defining a sentence is positive or negative.
- POS tagging → define the Part of speech of every word in a sentence, the output will also be a sequence.
- Natural Language Inference → given two sentences, and the model want to define if the two statements of the sentences contradict to each other.
- Extraction-Base question answering → given a document and a question (2 sentences), the output would be two integers i and j, the segment between ith word and jth word of the document should be the answer to the given question.
- Discuss the difference between BERT and distilBERT?
 - BERT is a large model with more than 100 million parameters, and distilBERT has only about 40% number of parameters compare with BERT but 60% faster than it. We can describe distilBERT as a BERT's student who tries to mimic the process and structure of BERT but with a smaller scale, distilBERT not only compute the loss between its output and ground truth during training, but it also computes the loss against BERT's forwarding output for back propagation. distilBERT makes it way easier for training process compare with BERT.
- Screenshot the required test F1-score.

```
batch 9900
Epoch: 0, F1 score: 0.9314, Precision: 0.9314, Recall: 0.9314, Loss: 0.2276
```

- (BONUS 5%) Explain the relation of the Transformer and BERT and the core of the Transformer. Application of deep learning 7-1 to 9-1 by Vivian
 - In brief, BERT can be seen as the encoder of Transformer, it read the whole word sequence at once, so it can learn the context of the word from the sequences in front of it and behind it. The core of the Transformer are encoder and decoder that both can be stacked. An encoder includes self-attention layer which apply self-attention

mechanism and feedforward layer that pass its output to the decoder, both the layers apply residual like structure and normalization for their output. A decoder also consists of the two layers mentioned above but with another layer that computes attention between encoder and decoder placed between the two layers. And the input of decoder is the ground truth during training, we want to get the same output from the probability distribution from the decoder.

- For Part 3
 - Briefly explain the difference between vanilla RNN and LSTM.
 - Though RNN learn from the information from the previous input, but it doesn't know what should be taken into consideration and what should be ignored, however, LSTM resolve this problem with a structure described as a conveyor belt which has the ability of adding and removing information through the entire chain.
 - Please explain the meaning of each dimension of the input and output for each layer in the model. For example, the first dimension of input for LSTM is batch size.

```
self.embedding = nn.Embedding(inputSize, embededDim, padding_idx=0)
self.LSTM = nn.LSTM(embededDim, hiddenSize, numLayer, dropout=0.4)
self.lin = nn.Linear(in_features=hiddenSize, out_features=2)
self.sig = nn.Sigmoid()
```

- embedding
 - input
 - first dimension → length of word sequence
 - second dimension → batch size
 - output
 - first dimension → length of word sequence
 - second dimension → batch size
 - third dimension → embedding dimension
- LSTM

- input
 - first dimension → length of word sequence
 - second dimension → batch size
 - third dimension → embedding dimension
- output
 - first dimension → length of word sequence
 - second dimension → batch size
 - third dimension → hidden size
- linear
 - input
 - first dimension → batch size
 - second dimension → hidden size
 - output
 - first dimension → batch size
 - number of classes (in our case is 2)
- Sigmoid
 - Sigmoid won't change the dimension of the input, input and output both have the same meaning as the output of linear layer for this part, however, we project every value in the input tensor into range [0,1] with this activation function.
 - Screenshot the required test F1-score.

```
Testing ...
Epoch: 10, F1 score: 0.8887, Precision: 0.8887, Recall: 0.8887, Loss: 0.8416
total batches: 5000
```

- Discussion
 - Discuss the innovation of the NLP field and your thoughts of why the technique is evolving from n-gram → LSTM → BERT.

- n-gram is the most intuitive model of the three. Even maybe if I encountered a relevant problem without knowing any model than my way of thought would be pretty close to n-gram, since it is basically based on some Probability and Statistics which is not too complicated and innovative, so this model should be come up with by somebody definitely early than the other two models. LSTM is based on RNN, and RNN is based on neural network, neural network is a pretty innovative idea though it has come out for centuries so as RNN, and LSTM is the evolution of RNN which has many mechanisms such as forget gate and conveyor belt, so it can be seen as the next level generation tool in NLP field. BERT is a self-supervised massive model related to Transformer structure, which means it also apply self-attention, which can solve many downsides in RNN structure models such as Parallelism and have even better performance. What's more, BERT use massive amount of data for its training process which most of our computing system can't afford, the number of parameters is also unimaginable. By unlabeled data, it enables it to learn rich contextualized representations of words, so it should be much stronger than LSTM. Then we get n-gram → LSTM → BERT. And nowadays, GPT models have Taken the world by storm.

- Describe problems you meet and how you solve them.

I spent unimaginable time doing part3.

At first. my loss doesn't decline as the epochs goes with RNN or LSTM, then I found out that it was cause by the fault I took the last output after linear layer as forward output, and the right way should be taking the last output of RNN layer and throw it into linear layer for classification, since the classifier should only consider the last output of RNN. After correcting this mistake, the performance stared to improve as the epoch goes, but I very soon came across another issue. The F1-score can never pass 80% no matter how I tuned those parameters, and then I realized it may because of the dimension of the forward output, which is one from sigmoid function, and it may not be enough to load that much information learned from the model, so I changed the output features to 2 which does

significantly improve the final result, I got the final result after tuning the hyperparameters larger and larger.

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]  Unzipping corpora/stopwords.zip.
strat training! mdfk
5000
batch 0 0
batch 1000 44114.9355995101929
batch 2000 3489.9838263191757
batch 3000 3344.09961285591
batch 4000 3254.351767063141
test
Epoch: 0, F1 score: 0.3333, Precision: 0.7500, Recall: 0.5000, Loss: 9.5000
strat training! mdfk
5000
batch 0 0
batch 1000 3124.817808866501
batch 2000 3029.7533301115036
batch 3000 3041.197437640085
batch 4000 3070.2252534627914
test
Epoch: 1, F1 score: 0.3333, Precision: 0.7500, Recall: 0.5000, Loss: 9.5000
strat training! mdfk
5000
batch 0 0
batch 1000 2956.182275056839
batch 2000 2989.0429618501663
batch 3000 2914.894508123398
batch 4000 2959.70296379898312
test
Epoch: 2, F1 score: 0.3333, Precision: 0.7500, Recall: 0.5000, Loss: 9.5000
strat training! mdfk
5000
batch 0 0
batch 1000 2890.488317012787
batch 2000 2894.5982773303986
batch 3000 2850.049678448624
batch 4000 2871.5756685247757
traceback (most recent call last):
  File "/content/main.py", line 240, in <module>
    second_part(model_type, df_train, df_test, N)
  File "/content/main.py", line 125, in second_part
    train
  File "/content/main.py", line 171, in train
```

RNN mistake

LSTM mistake

```

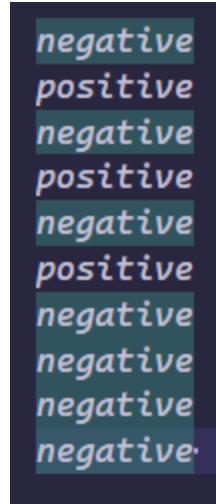
!export PYTORCH_CUDA_ALLOC_CONF=expandable_segments_size
!export PYTORCH_CUDA_ALLOC_CONF=max_split_size_mb
!python main.py --model_type RNN --preprocess 0

[nltk_data] Downloading package stopwords to /root/nltk_data
[nltk_data] Package stopwords is already up-to-date.
5000
batch 0 | loss: 0.0
batch 1000 | loss: 8.27168435072899
batch 2000 | loss: 8.27773313693436
batch 3000 | loss: 8.36663141116333
batch 4000 | loss: 8.326715202093125
test
Epoch: 0, F1 score: 0.3574, Precision: 0.5114, Recall: 0.2988
batch 0 | loss: 0.0
batch 1000 | loss: 8.271370239496231
batch 2000 | loss: 8.284134526968802
batch 3000 | loss: 8.342380714893341
batch 4000 | loss: 8.297759853124619
test
Epoch: 1, F1 score: 0.3937, Precision: 0.5403, Recall: 0.3000
batch 0 | loss: 0.0
batch 1000 | loss: 8.2789477780251502
batch 2000 | loss: 8.385661718821525
batch 3000 | loss: 8.277322221146355
batch 4000 | loss: 8.289455323457718
test
Epoch: 2, F1 score: 0.4394, Precision: 0.5307, Recall: 0.3000
batch 0 | loss: 0.0
batch 1000 | loss: 8.476744317770004
batch 2000 | loss: 8.16112879872322
batch 3000 | loss: 8.1675801653862
batch 4000 | loss: 8.317631958974464
test
Epoch: 3, F1 score: 0.4835, Precision: 0.5734, Recall: 0.3000
batch 0 | loss: 0.0
batch 1000 | loss: 8.23622814977169
batch 2000 | loss: 8.369783212065696
batch 3000 | loss: 8.391067562637329

```

after getting the last output from lstm but not linear layer, but gpu usage maximum was met,

- In-context learning (BONUS 5%)'
 - Try to use the LLMs (such as ChatGPT, Gemini, etc) with in-context learning to complete the task above. There are some keywords you can try: Few-shot, Chain of Thought, Socratic, etc. Please compare the results with different prompts and write down your findings in the report.
 - I input 8 set of (review, sentiment) first.
 - 10 tests set of review.
 - ground truth.



negative
positive
negative
positive
negative
positive
negative
negative
negative
negative

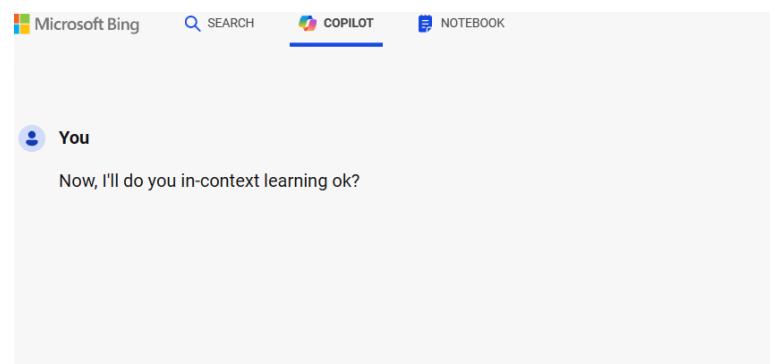
ChatGPT

1. Negative sentiment: The reviewer is disappointed by the film they saw at the Belgrade Film Festival, particularly because the second half was vastly different from the promising beginning. They criticize the film for being agonizingly long and painful to bear, with a lack of warning about its content.
2. Positive sentiment: The reviewer praises a film about four friends in the 60s, highlighting the strong direction, cast, and script. They appreciate how the film explores themes of growing up and father-son conflict, finding it both entertaining and insightful.
3. Negative sentiment: The reviewer criticizes a film starring Robert Montgomery and Myrna Loy, calling it highly predictable and contrived. They point out a scene with a bear as particularly unbelievable.
4. Positive sentiment: The reviewer enjoys a blaxploitation classic about a kung fu mama seeking revenge in Hong Kong. They find it great fun to watch, particularly praising the soundtrack and action scenes.
5. Mixed sentiment: The reviewer describes an adaptation of Shakespeare's "The Tempest" set in Mississippi during the Civil War. While they appreciate some aspects of the film, such as the powerful performance by Peter Fonda, they also criticize certain parts as comical or poorly explained.
6. Negative sentiment: The reviewer is disappointed by a film about a Viking warlord cursed to live violent lifetimes. They find the film boring and lacking in entertainment value, suggesting it could have been improved with more exploitation elements.
7. Negative sentiment: The reviewer is critical of an indie film called "Northfork," describing it as pretentious and devoid of meaning. They find the film's direction confusing and uninteresting, ultimately recommending viewers to avoid it.
8. Negative sentiment: The reviewer expresses disappointment with a film they expected to be entertaining but found boring and poorly executed. They criticize the film for not living up to its intriguing premise and suggest it could have been improved with more exploitation elements.
9. Negative sentiment: The reviewer is highly critical of a film, describing it as bad and boring. They question the film's audience and criticize its predictable plot and overacted characters.
10. Negative sentiment: The reviewer pans an apocalyptic zombie film, calling it lame and comparing it to a high school play gone bad. They criticize the acting, special effects, and overall quality of the film, advising others not to watch it.

ChatGPT, accuracy → 100%

1. Negative - Dislikes the second half of the film, finding it painful and drawn-out.
2. Positive - Praises the film's look, direction, cast, script, and themes.
3. Negative - Finds the film predictable and contrived.
4. Positive - Enjoys the film for what it is, praising the acting, soundtrack, and genre.
5. Mixed - Finds some aspects interesting but criticizes others for being unclear or unnecessary.
6. Positive - Portrays the film in a positive light, highlighting the action and revenge themes.
7. Negative - Disappointed by the film, finding it boring and unfulfilling.
8. Negative - Scathing review, criticizing the film for being pretentious, boring, and offering nothing new.
9. Negative - Hates the film, calling it terrible and boring.
10. Negative - Dislikes the film, criticizing the acting, plot, and special effects.

Gemini, accuracy → 100%



Microsoft COPILOT can't even response

Here are the sentiments for the 10 reviews:

1. Negative
2. Positive
3. Negative
4. Positive
5. Mixed
6. Positive
7. Negative
8. Extremely Negative
9. Extremely Negative
10. Negative

CLAUDE, accuracy → 100%, even with more detailed classification

- my findings → I try three models successfully.
 - ChatGPT → worst performance of the three in my opinion, need to make my question clearly to make it understand what it should do, and the output is not clean and formatted, which looks quite annoying.
 - Gemini → more intelligent than ChatGPT that can understand your requirement instantly and perform tidy and formatted output with only few essential key points informed for each review, very fast and clean.
 - CLAUDE → remarkable UI/UX experience through the whole process that everyone should try, the prompt windows keep switching as the topic go deeper, I have never seen LLM interface like this before. What's more eye-catching is that it even extended the class number to more than two without asking to do so as we can see above, and the output is super clean with just the

prediction I requested, outperform the other two in some perspectives.

The screenshot shows a dark-themed conversational AI interface. At the top, a large white text area displays the message: "Good evening, Now, I'll do you in-context learning". Below this is a input field with placeholder text "What can I help you with?" and a "Start Chat" button with a right-pointing arrow. A list item "• Sentiment: Negative" is shown. Underneath, a "Pasted Text - 11 lines" section contains a preview of 11 lines of text from a movie review. A blue "PASTED" button is visible. On the left, there's a small circular icon with "NI". To the right, a list titled "Here are the sentiments for the 10 reviews:" is displayed, containing the following items:

- 1. Negative
- 2. Positive
- 3. Negative
- 4. Positive
- 5. Mixed
- 6. Positive
- 7. Negative
- 8. Extremely Negative

- In my opinion, CLAUDE > Gemini > ChatGPT in my n-context learning experience this time.