Imperial College Business School

Machine Learning and Finance -Solution-

Time allowed: 2 hours

Description of the exam:

The exam is graded on a 100-point scale. It is divided into two independent problems.

- **Problem A:** Credit Risk Prediction (30 marks).
- **Problem B:** Attention Based Bi-LSTM Sentiment Analysis Model (70 marks).

The following table shows the marks for each of the 27 questions across 8 sections.

Problem	Section	Question	Number of marks
A.	1. Preliminary Questions	1	3
		2	3
		3	3
	2. The Random Forest Classifier	4	3
		5	4
		6	4
		7	3
		8	2
		9	5
B.	3. Introducing the Problem	10	3
	4. The Preprocessing Step	11	3
		12	3
	5. The Prediction Model	13	4
		14	3
		15	3
		16	4
		17	3
		18	3
	6. The Forward Propagation	19	4
		20	4
		21	4
	7. The Optimization Process	22	3
		23	3
		24	3
		25	4
		26	4
	8. Replacing the LSTM with a Self Attention Model	27	8
		28	4

Problem A: Credit Risk Prediction

In this section, our goal is to develop a model that can predict the quality of loans based on $D = 15$ different features:

- 10 numerical features denoted: $f_1^N, f_2^N, \ldots, f_{10}^N$.
- 5 categorical features denoted: $f_1^C, f_2^C, \ldots, f_5^C$.

Each D-dimensional feature vector corresponds to a target reflecting the level of risk.

There are two possible categories: 0 indicates a low level of risk, while 1 indicates a high level of risk.

Using a supervised learning algorithm, we would like to learn a mapping function from the feature space to the target space.

The dataset is composed of $N_1 = 1000$ training samples $(X_i, y_i)_{1 \leq i \leq N_1}$ and N_2 testing samples $(\tilde{X}_i, \tilde{y}_i)_{1 \leq i \leq N_2}$.

The dataset is highly imbalanced since 90% of the training feature vectors are associated with a target 0 (i.e., low level of risk), while 10% are associated with a target 1 (i.e., high level of risk).

1 Preliminary questions

Question 1: What makes the accuracy score unsuitable as an evaluation metric for this specific problem?

When dealing with an imbalanced dataset, a naive model falsely predicting the majority class will have a high accuracy without being a good model. For instance, a model always predicting the target 0 will have an accuracy score of 90% is our case. We should prefer other evaluation metrics like the F1 score or the AUC.

Table 1 summarizes the number of possible categories for each categorical variable.

Categorical Variable	Number of possible categories
f_1^C	4
f_2^C	3
f_3^C	6
f_4^C	2
f_5^C	7

Table 1

We would like to use the one hot encoding processing strategy to encode all the categorical variables.

Question 2: What is the new shape of the training dataset after it has been processed ?

The number of numerical features is 10. Each categorical feature having K possible categories is gonna be transformed into $K - 1$ dummy variables. The total number of dummy variables is: $3 + 2 + 5 + 1 + 6 = 17$ The new shape of the training dataset is: $(N_1, 27)$ for the features and $(N_1, 1)$ for the targets.

The categorical feature f_1^C can take 4 possible categories $\{A, B, C, D\}$.

Table 2 represents the values of the categorical variable f_1^C for the first three training samples.

Index of the training sample	...	categorical feature f_1^C	...
0	...	B	...
1	...	A	...
2	...	C	...

Table 2

Question 3: What would be the processed representation of the column f_1^C for the first 3 training samples ?

We use the one hot encoding process to transform each category into a 4-dimensional vector. Then, we drop the first dimension. We get the following representation:

Index of the training sample	Dummy_variable_1	Dummy_variable_2	Dummy_variable_3
0	1	0	0
1	0	0	0
2	0	1	0

2 The Random Forest Classifier

We wish to train a Random Forest classifier on the training dataset.

Question 4: Give 3 hyperparameters associated with the Random Forest classifier ?

- **The impurity measure** (the entropy, the gini, etc.)
- **The number of trees.**
- **The depth of each tree.**

Question 5: Explain how to optimize the hyperparameters using Grid Search and cross validation.

- First, we choose the set of N_h different combinations of hyperparameters (indexed $1, \dots, N_h$) we wish to test and the evaluation metric we would like to optimize. In our case, we can choose the F1 score.
- We divide the training dataset into K folds $(\mathcal{F}_1, \dots, \mathcal{F}_K)$.
- For each combination $i \in \{1, \dots, N_h\}$ of hyperparameters:
  - For each $k \in \{1, \dots, K$, we train the classifier on the dataset composed of the K-1 folds $(\mathcal{F}_1, \dots, \mathcal{F}_{k-1}, \mathcal{F}_{k+1}, \dots, \mathcal{F}_K)$ and we test it on the $\mathcal{F}_k$ fold. We end up with a performance f_k^i.
  - The performance associated with the combination of hyperparameters i is the average of the performances f_k^i:

$$f^i = \frac{1}{K} \sum_{k=1}^{K} f_k^i$$

- The best combination of hyperparameters is the one the highest performance f^i.

Let us consider one of the decision trees $(\mathcal{D})$ composing the Random Forest classifier.

Table 3 represents the values of an attribute and the target at a particular node of $(\mathcal{D})$. The node contains 10 samples.

...	Attribute	...	Target
...	1	...	0
...	1	...	0
...	0	...	1
...	0	...	1
...	0	...	1
...	1	...	0
...	1	...	0
...	0	...	1
...	1	...	0
...	0	...	1

Table 3

Question 6: Calculate the information gain associated with a splitting strategy on the attribute represented in table 3 ?

For a Bernoulli distrbution with a parameter p, the entropy is 1 when $p = 0.5$ and 0 when $p \in \{0, 1\}$.

$$\begin{aligned} \text{IG} &= H(Y) - 0.5H(Y|\text{Attribute} = 0) - 0.5H(Y|\text{Attribute} = 1) \\ &= 1 - 0.5 * 0 - 0.5 * 0 \\ &= 1 \end{aligned}$$

Question 7: If the depth of the decision tree allows further splitting steps, do you think we should split on this attribute?

As the entropy of a Bernoulli distribution is $\in [0, 1]$. The maximum value for the information gain is 1. Therefore, we should split on the Attribute defined in Question 6 since it has the maximum information gain.

After training the optimal Random Forest model, we got the ROC curve represented in figure 1. The red dot represents the default threshold $\tau_0 = 0.5$

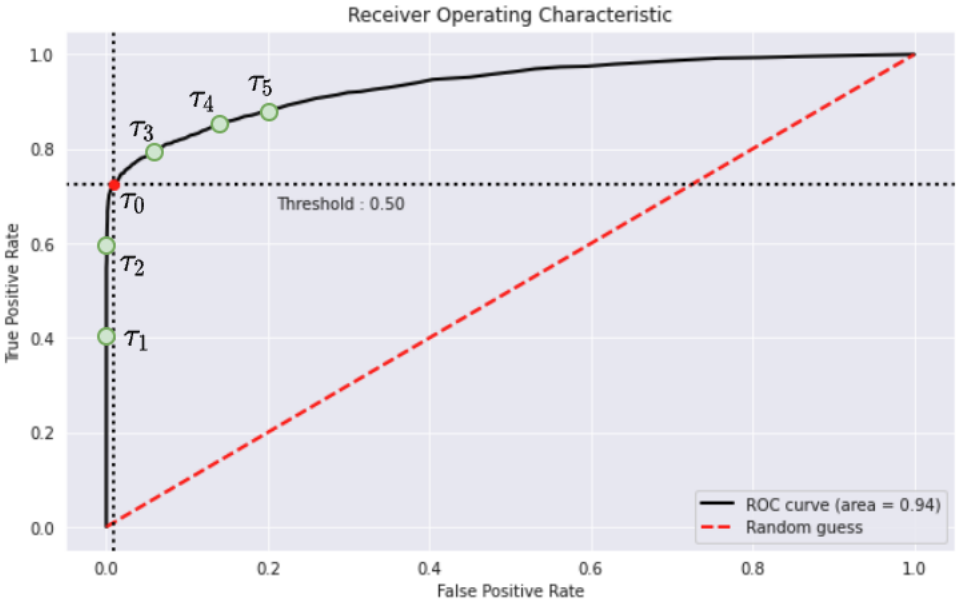

Figure 1: The ROC curve for the optimal Random Forest Classifier

Question 8: What is the value of the AUC ?

The AUC is defined as the area under the ROC curve. AUC = 0.94 according to figure 1.

Table 4 represents the True Positive Rate (TPR) and the False Positive Rate (FPR) associated with the thresholds τ_3, τ_4, τ_5 represented in figure 1.

Threshold	TPR	FPR
τ_3	0.8	0.05
τ_4	0.85	0.12
τ_5	0.9	0.2

Table 4

Question 9: Which of the thresholds $\tau_1, \tau_2, \tau_3, \tau_4, \tau_5$ would you use if your objective is to have the best F1-score with the constraint of a recall greater or equal to 0.8 ? Justify tour answer.

Among the thresholds $\tau_1, \tau_2, \tau_3, \tau_4, \tau_5$, only τ_3, τ_4, τ_5 have a recall greater or equal to 0.8. We know that the total number of training data is $N_1 = 1000$, 90% of them are positive. We deduce TP+FN = 900 and FP+TN = 100. We also know that $TP = TPR * (TP + FN)$ and $FP = FPR * (FP + TN)$. We can then deduce the precision Precision = $\frac{\text{TP}}{TP+FP}$. We already have the Recall (since the TPR is given). We deduce the F1 score

$$\text{F1 score} = 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The following table summarizes all the steps:

Threshold	**TPR**	**FPR**	TP	FP	Precision	F1 Score
-	-	-	$\text{TPR} * (0.9 * N_1)$	$\text{FPR} * (0.1 * N_1)$	$\frac{\text{TP}}{\text{TP+FP}}$	$2\frac{\text{Precision} \times \text{Recall}}{\text{Precision+Recall}}$
τ_3	0.8	0.05	720	5	720/725	0.88
τ_4	0.85	0.12	765	12	765/777	0.94
τ_5	0.9	0.2	810	20	810/830	0.93

The best threshold is τ_4, it has the best F1 score (0.94) with a recall of (0.85), which is greater than 0.8.

Problem B: Attention Based Bi-LSTM Sentiment Analysis Model

3 Introducing the problem

In this section, we wish to create a sentiment analysis model for daily financial news associated with a universe of $N_u = 500$ stocks from 2010 to 2021.

Figure 2 represents the splitting strategy into a training period of T_{train} days from the beginning of 2010 to the end of 2014 and a testing period of T_{test} days from the beginning of 2015 to the end 2021.

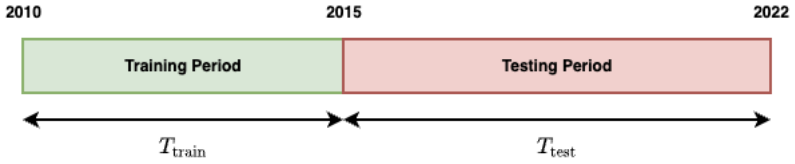

Figure 2: Training - Testing Periods

We make the assumption that the universe of stocks remains the same for the whole period.

Each stock is associated with a unique Id. Therefore, the stocks are represented in the dataset by a column named **StockId**, which takes values in $\{1, \ldots, N_u\}$.

There are N news in the training period. Each news is also associated with a unique Id in $\{1, \ldots, N\}$.

Similarly, there are N' news in the testing period represented with a unique Id in $\{1, \ldots, N'\}$.

These Ids are represented by a column named **NewsId** in both the training dataset ($\mathcal{D}_{\text{train}}$) and the testing dataset ($\mathcal{D}_{\text{test}}$).

For each date t in the training or the testing period, for each stock of id $i \in \{1, \ldots, N_u\}$, there are $n_{t,i}$ news.

Each news in the training dataset is associated with a target called **Sentiment**. There are $K = 3$ possible targets: **negative** (represented by the integer 0), **neutral** (represented by the integer 1) and **positive** (represented by the integer 2).

Table 5 represents the news associated with a specific date t in the training period and a specific stock of id i. This part of the training dataset contains $n_{t,i}$ news indexed from k to $k + n_{t,i} - 1$.

NewsId	**date**	**StockId**	**News**	**Sentiment**
⋮	⋮	⋮	⋮	⋮
k	t	i	news of index k	0
⋮	⋮	⋮	⋮	⋮
$k + n_{t,i} - 1$	t	i	news of index $k + n_{t,i} - 1$	2
⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮

Table 5

Question 10: What is the total number of training samples N (i.e, the total number of rows in the training dataset) as a function of $(n_{ti})_{\substack{1\leq t\leq T_{\mathbf{train}} \\ 1\leq i\leq N_u}}$?

The total number of training samples is:

$$N = \sum_{t=1}^{T_{\text{train}}} \sum_{i=1}^{N_u} n_{t,i}$$

We would like to learn from the training dataset a mapping function from the news space to the target space. This mapping function, parameterized by θ and denoted Φ_θ is represented in the figure 3.

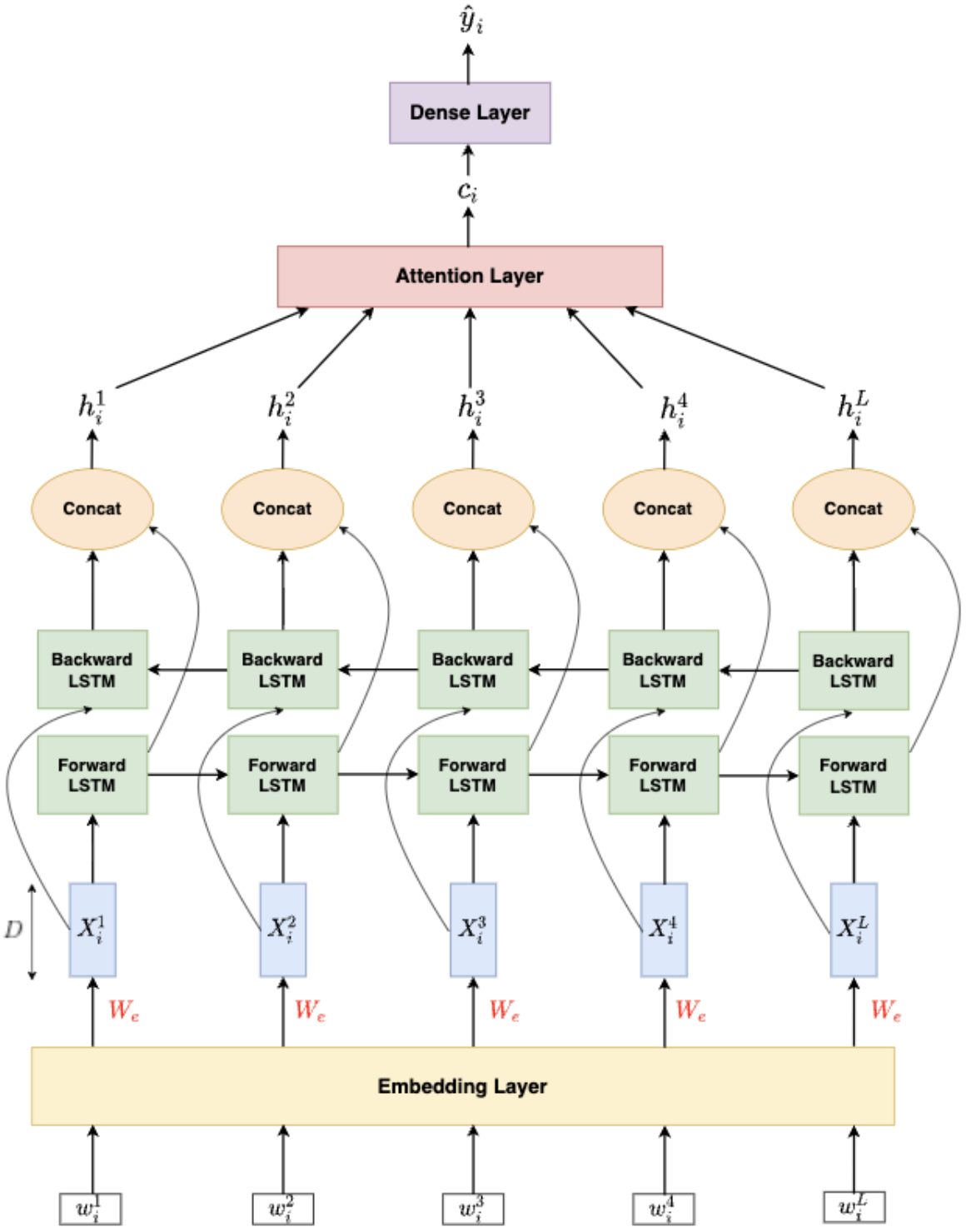

Figure 3: The Bi-LSTM Attention Based Sentiment Analysis Model

Section 4 describes how to transform the news into lists of integers of the same size.

4 The Preprocessing Step

As a first processing step, we would like to turn all the N news in the training dataset into lists of integers of the same size L, where each integer takes values in $\{0, \dots, V-1\}$ (V represents the vocabulary size).

For instance, we would like to encode the news of id $i \in \{1, \dots, N\}$ into a sequence of integers $w_i = (w_i^1, \dots, w_i^L)$,

as shown in table 6.

NewsId	**News**
1	$w_1 = (w_1^1, \dots, w_1^L)$
$\vdots$	$\vdots$
i	$w_i = (w_i^1, \dots, w_i^L)$
$\vdots$	$\vdots$
N	$w_N = (w_N^1, \dots, w_N^L)$

Table 6

Question 11: Explain how to convert all the lists of news into the sequences $w_i = (w_i^1, \dots, w_i^L)$ for $i \in \{1, \dots, N\}$.

This preprocessing step is composed of two parts:
- The first part is to transform the news into lists of integers of different size. For that, we need to to define a dictionary mapping each word to a unique index.
- The second step is to transform these lists of integers of different sizes into lists of integers of the same size L. For that, we keep the first L indices of each processed list if its length is $\geq L$. Otherwise, we add zeros to the list in order to reach the size L. That's why the index 0 is reserved for padding.

Question 12: What is the shape of the training dataset after it has been processed ?

The shape of the processed training dataset is (N, L)

5 The Prediction Model

For this whole section, let us consider a processed sequence $w_i = (w_i^1, \dots, w_i^L)$.

We would like to describe the forward propagation in order to get the final prediction $\hat{y}_i = \Phi_\theta(w_i) = \Phi_\theta(w_i^1, \dots, w_i^L)$.

5.1 The Embedding Layer

We would like to use pre-trained word embedding vectors of dimension $D = 200$ and freeze the embedding matrix during the optimization process.

Question 13: Describe briefly an algorithm of your choice that we can use to get the embedding vectors.

- The **GloVe** algorithm, which consists of a weighted least squares model that trains on global word-word co-occurence counts.
- The **Wod2vec** algorithms, a window based model, which learns word embeddings by making predictions in local context windows. The intuition behind the model is summarized by the famous quote "You shall know a word by the company it keeps". The model demonstrates the capacity to capture complex linguistic patters beyond word similarity.

Let W_e be the embedding matrix derived from the pre-trained word embeddings.

Question 14: What is the shape of W_e ?

The shape of W_e is (V, D).

Let $X_i = (X_i^1, \dots, X_i^L)$ be the sequence of the embedding vectors associated with w_i.

Question 15: For each $l \in \{1, \dots, L\}$, how can we get X_i^l from w_i^l and W_e ?

X_i^l is the row of index w_i^l of W_e

5.2 The Bi-LSTM layer

We use a bidirectional LSTM so as to obtain the hidden features $(h_i^1, \dots, h_i^L)$.

The model is composed of two LSTM layers, processing the embedding vectors in both directions:

- The **Forward LSTM** processes the embedding vectors from X_i^1 to X_i^L.
- The **Backward LSTM** processes the embedding vectors from X_i^L to X_i^1.
- Both LSTMs have hidden states of size M.

The hidden states associated with the Forward LSTM are denoted $(f_i^l, C_i^l)_{1 \leq l \leq L}$.

Figure 4 represents the process of generating the new hidden states (f_i^l, C_i^l) from the previous hidden states (f_i^{l-1}, C_i^{l-1}) and the new embedding vector X_i^l.

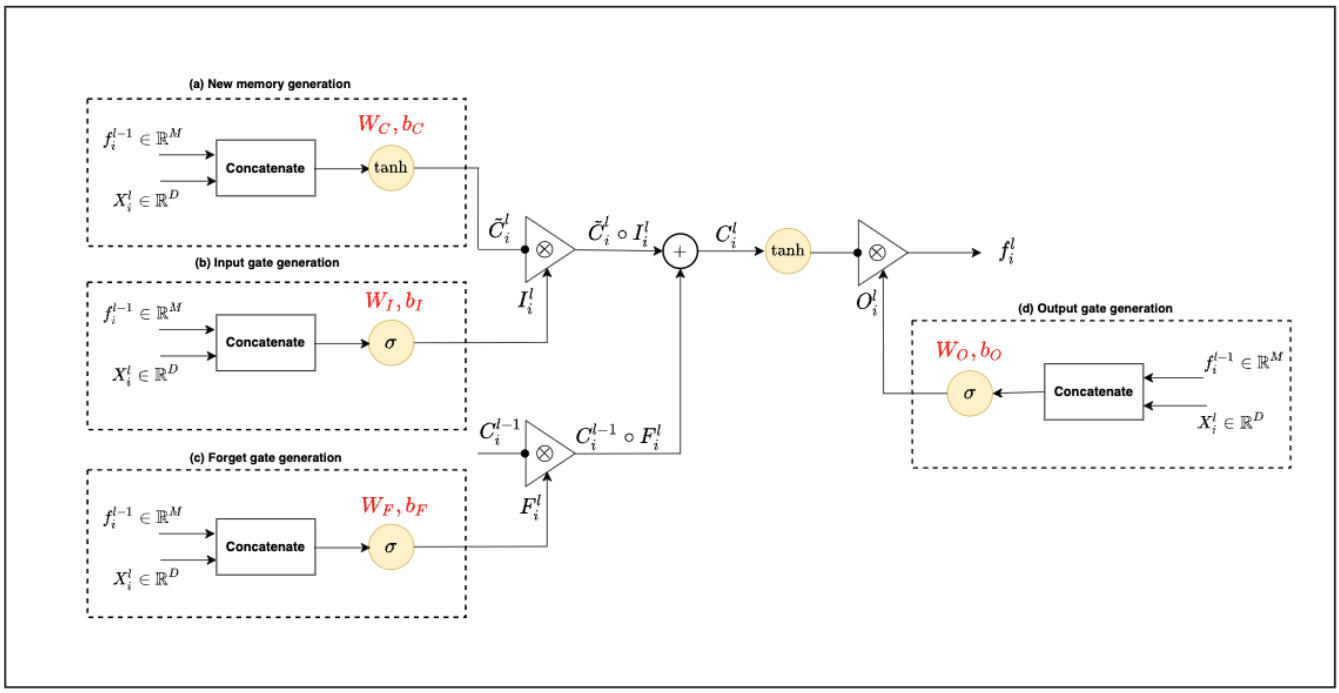

Figure 4: A representation of the Forward LSTM Layer

Similarly, the hidden states of the Backward LSTM are denoted $(b_i^l, \hat{C}_i^l)_{1 \leq l \leq L}$.

The final hidden states $(h_i^1, \ldots, h_i^L)$ are calculated as follows:

$$\forall l \in \{1, \ldots, L\} \quad h_i^l = f_i^l \oplus b_i^l \quad \text{Where } \oplus \text{ denotes the function of concatenation}$$

Question 16: Give the expression of the new hidden states (f_i^l, C_i^l) as a function of $(f_i^{l-1}, C_i^{l-1}, X_i^l)$ and the parameters of the Forward LSTM layer $(W_I, b_I), (W_F, b_F), (W_O, b_O), (W_C, b_C)$.

$$\begin{aligned}
F_i^l &= \sigma\left(W_F^T[f_i^{l-1} \oplus X_i^l] + b_F\right) \\
I_i^l &= \sigma\left(W_I^T[f_i^{l-1} \oplus X_i^l] + b_I\right) \\
\tilde{C}_i^l &= \sigma\left(W_C^T[f_i^{l-1} \oplus X_i^l] + b_C\right) \\
C_i^l &= \tilde{C}_i^l \circ I_i^l + C_i^{l-1} \circ F_i^l \\
O_i^l &= \sigma\left(W_O^T[f_i^{l-1} \oplus X_i^l] + b_C\right) \\
f_i^l &= O_i^l \circ \tanh(C_i^l)
\end{aligned}$$

5.3 The Attention Layer

We would like to add an attention layer in order to find the contribution of each word in the process of generating the final prediction.

For each $l \in \{1, \dots, L\}$, the attention mechanism should assign a weight α_i^l to each hidden state h_i^l.

The weights $(\alpha_i^l)_{1 \leq l \leq L}$ should be positive and should sum to 1.

$$\forall l \in \{1, \dots, L\} \quad \alpha_i^l \geq 0 \quad \text{and} \quad \sum_{l=1}^{L} \alpha_i^l = 1$$

The first step in processing the hidden states $(h_i^1, \dots, h_i^L)$ using the attention layer is to produce a score $s_i^l \in [-1, 1]$ associated with each h_i^l for all $l \in \{1, L\}$ as follows:

$$\forall l \in \{1, \dots, L\} \quad s_i^l = \tanh\left(W_a^T h_i^l + b_a\right) \quad \text{with} \quad W_a \in \mathbb{R}^{2M \times 1}, b_a \in \mathbb{R}$$

Question 17: For all $l \in \{1, \dots L\}$, give the expression of α_i^l as a function of the scores $(s_i^{l'})_{1 \leq l' \leq L}$.

$$\alpha_i^l = \frac{e^{s_i^l}}{\sum_{l'=1}^{L} e^{s_i^{l'}}}$$

The output of the attention layer can then be expressed as follows:

$$c_i = \sum_{l=1}^{L} \alpha_i^l h_i^l$$

5.4 The Final Dense Layer

We use a final dense layer to turn c_i into the final prediction $\hat{y}_i \in \mathbb{R}^K$.

Question 18: What should be the activation function of the final dense layer ?

As we are dealing with a multiclass classification problem, the final activation function should be the **softmax** activation function.

6 The Forward Propagation

Let us consider a batch $(\mathcal{B})$ of size N_b, table 7 represents the shapes of the different tensors after applying the different layers of the Forward Propagation to the batch $(\mathcal{B})$.

The Layer	The Shape of the output Tensor	The number of parameters
Input Layer	t_1	p_1
Embedding Layer	t_2	p_2
Bidirectional LSTM Layer	t_3	p_3
Attention Layer	t_4	p_4
Dense Layer	t_5	p_5

Table 7

Question 19: Give the expressions of the tuples $(t_1, t_2, t_3, t_4, t_5)$ representing the shapes of the output tensors after applying the corresponding layers in the first column of table 7.

$$t_1 = (N_b, L)$$
$$t_2 = (N_b, L, D)$$
$$t_3 = (N_b, L, 2M)$$
$$t_4 = (N_b, 2M)$$
$$t_5 = (N_b, K)$$

Question 20: Give the expressions of the number of parameters $(p_1, p_2, p_3, p_4, p_5)$ parameterizing the corresonding layers in the first column of table 7 as a function of V, D, M and K.

$$p_1 = 0$$
$$p_2 = V * D$$
$$p_3 = 8 * [M * (D + M) + M]$$
$$p_4 = 2M + 1$$
$$p_5 = 2M * K + K$$

(As a reminder, V is the vocabulary size, D is the embedding dimension, M is the size of the hidden vectors for each LSTM layer and K is the number of possible targets).

Question 21: By setting the hyperparameters to the following values $V = 10000, D = 200, M = 64$, calculate the total number of trainable parameters and the total number of non trainable parameters.

- The number of trainable parameters is $p_2 = 2000000$
- The number of non trainable parameters is $p_3 + p_4 + p_5 = 135680 + 129 + 387 = 136196$

7 The optimization process

We would like to optimize the trainable parameters using an appropriate optimization algorithm. We divide the training samples into training and validation.

Question 22: Which loss should we use in order to optimize the parameters of the model Φ_θ ?

As we are dealing with a multiclass classification problem, we should use the **categorical crossentropy loss**.

Question 23: Describe an appropriate optimization algorithm with an adaptive learning rate

- We can use the Adam optimizer (Adaptive Moment Estimation), which computes adaptive learning rates for each parameter.
- In addition to storing an exponentially decaying average of past squared gradients v_t, Adam also keeps an exponentially decaying average of past gradients m_t

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

- As m_t and v_t are initialized as vectors of zeros, they are biased towards zero during the initial time steps.
- Thus, we use the bias-corrected first and second moment estimates:

$$\tilde{m}_t = \frac{m_t}{1-\beta_1^t}$$
$$\tilde{v}_t = \frac{v_t}{1-\beta_2^t}$$

- We fix the learning rate η
- The Adam update equation of the parameters θ is then:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\tilde{v}_t} + \epsilon} \tilde{m}_t$$

- $\epsilon = 10^{-8}$ is a very small value used to avoid dividing by zero.
- We usually use $\beta_1 = 0.9$ and $\beta_2 = 0.999$.
- We can see that in the update equation, the learning rate depends on $\tilde{v}_t$.

Figure 5 represents the training and validation loss for 50 epochs.

Figure 5: The training and validation loss

Question 24: What is the problem highlighted in the figure 5 ?

It's an **overfitting** problem since the validation loss starts decreasing after few epochs.

Question 25: Give two ways of overcoming the aforementioned problem.

- **Dropout**: It consists in randomly dropping out a number of output features of the layer during training.
- **Weight regularization**: It consists in adding to the loss function a cost associated with having large weights. (The $\mathcal{L}^1$ norm of the weights or the $\mathcal{L}^2$ norm can be used as penalty terms).
- **Early stopping**: which consists in stopping the training process when the validation loss starts increasing.

We compared the Attention based Bi-LSTM model described in figure 3 with a regular LSTM model, we obtained the confusion matrices on the validation set (figure 6):

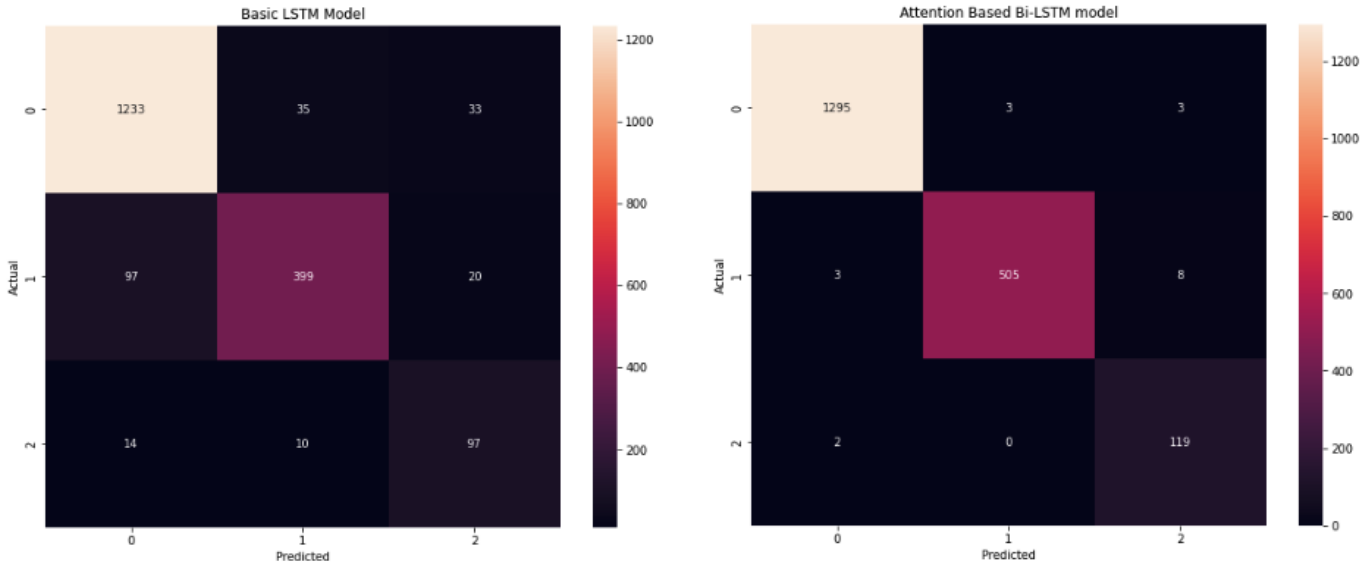

Figure 6: The confusion matrix on the validation set for both models

Question 26: Which model would you prefer ? Justify your answer.

The Attention Based Bi-LSTM model has lower errors for all the elements of the confusion matrix (i.e, the element outside the main diagonal). Therefore, The Attention Based Bi-LSTM model should be preferred.

8 Replacing the LSTM with a Self Attention Model

Question 27: Propose a new architecture to perform the same task using a self attention layer instead of the Bi-LSTM layer. Describe all the layers involved in the new architecture and the shapes of the tensors after each layer transformation.

Figure 7 represents the new architecture based on the Attention mechanisms.

- The first step is the same as before. We start with a sequence of integers $w_i = (w_i^1, \dots, w_i^L)$ which we transform into embedding vectors $(X_i^1, \dots, X_i^L)$.
- The second step is an iteration of N_L times the following transformations:
  - We use H heads in the MultiHeadAttention Layer.
  - The output is denoted O_1, it is obtained as follows:

$$
\begin{aligned}
&O_1 = \text{Concat}(\text{head}_1, \dots, \text{head}_H) W_O \\
&\forall h \in \{1, \dots, H\} \quad \text{head}_h = \text{Attention}\left(QW_h^Q, KW_h^K, VW_h^V\right) \\
&\text{with} \quad \text{Attention}(Z_1, Z_2, Z_2) = \text{Softmax}(\frac{Z_1 Z_2^T}{\sqrt{D}}) Z_3
\end{aligned}
$$

  - We add a residual connection with a normalization layer $\mathcal{L}_1$.
  - The result O_2 is obtained as follows:

$$O_2 = \mathcal{L}_1 (O_1 + X)$$

  - The next step is a basic Feed Forward Neural Network.
  - The output is denoted O_3
  - We use another normalization layer $\mathcal{L}_2$.
  - The result O_3 is obtained as follows:

$$O_3 = \mathcal{L}_2 (O_3 + O_2)$$

- After N_L iteration, the output is the sequence $h_i^1, \dots, h_i^L$.
- We can then use the same layers as in figure 3: an Attention layer:

$$
\begin{aligned}
\forall l \in \{1, \dots, L\} \quad & s_i^l = \tanh(W_a^T h_i^l + b_a) \\
\forall l \in \{1, \dots, L\} \quad & \alpha_i^l = \frac{e^{s_i^l}}{\sum_{l'=1}^{L} e^{s_i^{l'}}} \\
& c_i = \sum_{l=1}^{L} \alpha_i^l h_i^l \\
& \hat{y}_i = \text{Softmax}(W_d^T c_i + b_d)
\end{aligned}
$$

For a particular batch $\mathcal{B}$ of size N_b, table 8 summarizes all the layers and the corresponding output shapes.

The layer	**The Shape of the output Tensor**
Input Layer	(N_b, L)
Embedding Layer and Positional Encoding	(N_b, L, D)
Self Attention Layer	(N_b, L, D)
Attention Layer	(N_b, D)
Dense Layer	(N_b, K)

Table 8

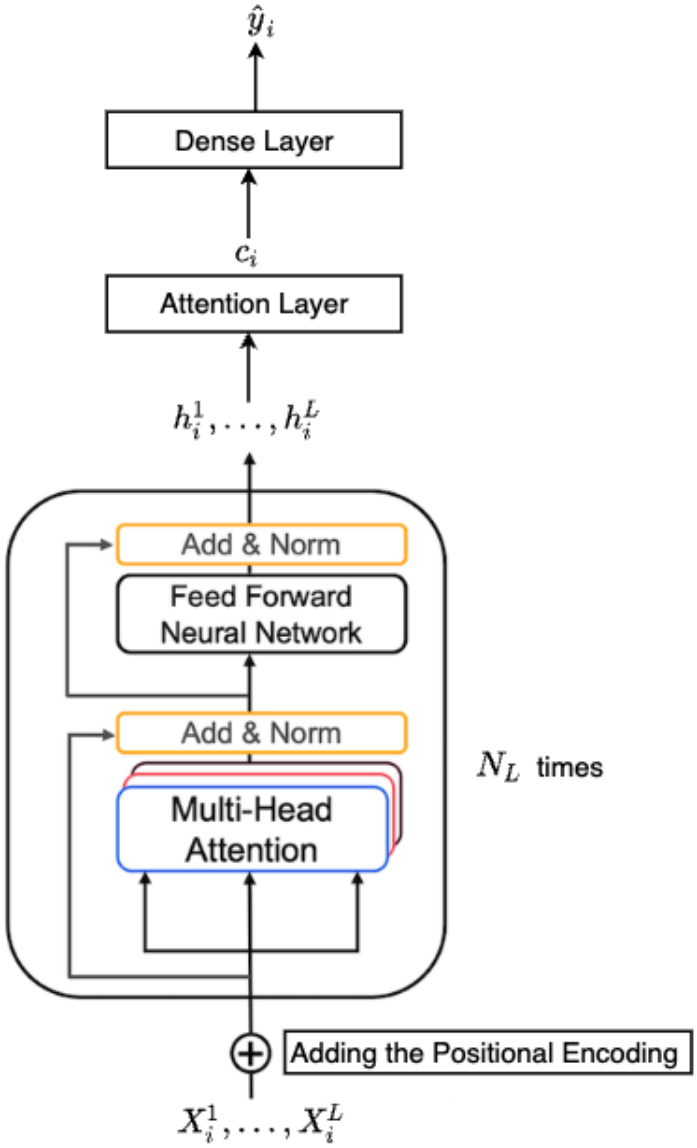

Figure 7: Self Attention architecture for Sentiment Analysis

During the testing period 2015 - 2021, we intend to use the sentiment analysis model to forecast the sentiment associated with the news attributed to each stock in the universe. The goal of this exercise is to select the best-performing stocks (Category A) and the worst-performing stocks (Category C).

This figure 8 shows the evolution of three portfolios equally weighting the stocks in categories A and C, as well as the entire universe during the testing period 2015-2021.

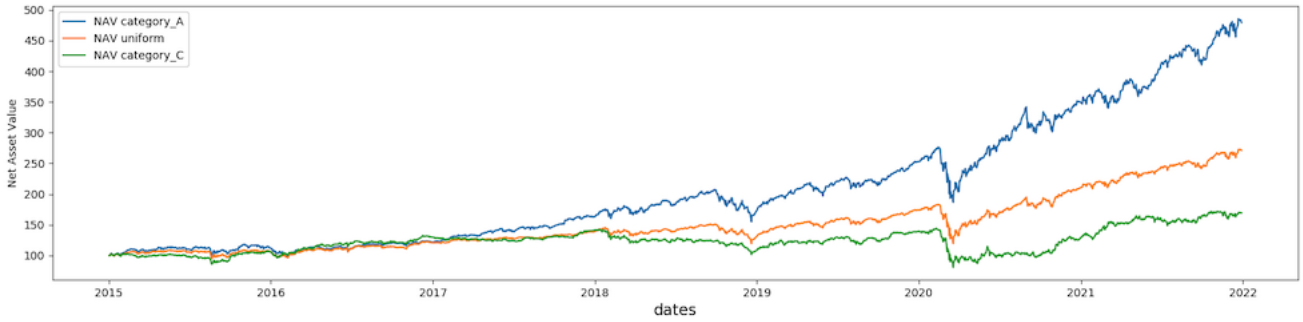

Figure 8: The Evolution of the 3 portfolios: Category A - Category C - Whole universe

Question 28: Describe how the trained Sentiment Analysis Model can be used to create the aforementioned stock picking strategy.

Let t be a date in the testing period and $i \in \{1, \dots, N_u\}$ an id representing a stock of the universe.
There are n_{ti} news corresponding to date t and id i, indexed $k, \dots, k + n_{ti} - 1$.
Using the Sentiment Analysis Model, we can get the predictions $\hat{y}_k, \dots, \hat{y}_{k+n_{ti}-1}$.
Let us consider $k' \in \{k, \dots, k + n_{ti} - 1\}$.
We turn $\hat{y}_{k'}$ into an integer $\tilde{y}_{k'} \in \{0, 1, 2\}$ reflecting the sentiment of the news of id k'.

$$\forall k' \in \{k, \dots, k + n_{ti} - 1\} \quad \tilde{y}_{k'} = \underset{d \in \{0,1,2\}}{\arg\max}\, \hat{y}_{k'}[d]$$

We can then calculate a score ξ_t^i associated with the stock of id i for date t as follows:

$$\xi_t^i = \frac{\sum_{k'=k}^{k+n_{ti}-1} 1_{\{\tilde{y}_{k'}=2\}} - \sum_{k'=k}^{k+n_{ti}-1} 1_{\{\tilde{y}_{k'}=0\}}}{\sum_{k'=k}^{k+n_{ti}-1} 1_{\{\tilde{y}_{k'}=2\}} + \sum_{k'=k}^{k+n_{ti}-1} 1_{\{\tilde{y}_{k'}=0\}}}$$

For each date t, we can then decompose the scores associated with the whole universe $\xi_t^1, \dots, \xi_t^{N_u}$ into tertiles defining 3 clusters of stocks (Category A, Category B, Category C).