Language translation

NIV KOTEK – 208236315

SAEED ESAWI - 314830985

Our Github repo for this project can be found here.
The original source repo for this project is located here.

Moderator: OR HAIM ANIDJAR

Introduction

Project Goal:

In this project, we build a deep neural network that functions as part of a machine translation pipeline. The pipeline accepts English text as input and returns the French translation. The goal is to achieve the highest translation accuracy possible.

Why Machine Translation Matters:

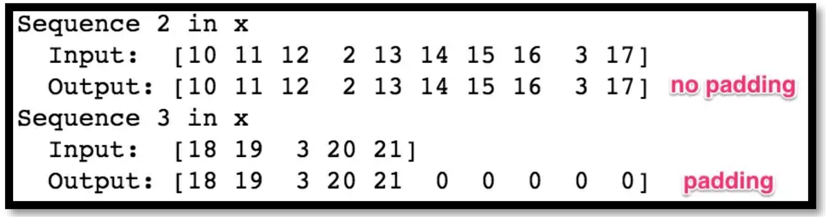
The ability to communicate with one another is a fundamental part of being human. There are nearly 7,000 different languages worldwide. As our world becomes increasingly connected, language translation provides a critical cultural and economic bridge between people from different countries and ethnic groups.

Some of the more obvious use-cases include:

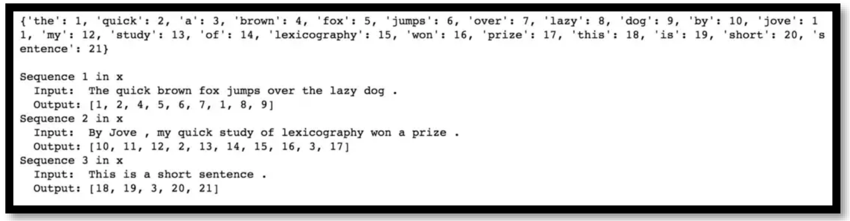
- business: international trade, investment, contracts, finance
- commerce: travel, purchase of foreign goods and services, customer support
- media: accessing information via search, sharing information via social networks, localization of content and advertising
- education: sharing of ideas, collaboration, translation of research papers
- government: foreign relations, negotiation

Today, Google and Microsoft can translate over 100 different languages and are approaching human-level accuracy for many of them.

Introduction



```
Sequence 2 in x
  Input:  [10 11 12  2 13 14 15 16  3 17]
  Output: [10 11 12  2 13 14 15 16  3 17]
Sequence 3 in x
  Input:  [18 19  3 20 21]
  Output: [18 19  3 20 21  0  0  0  0  0]
```



```
{'the': 1, 'quick': 2, 'a': 3, 'brown': 4, 'fox': 5, 'jumps': 6, 'over': 7, 'lazy': 8, 'dog': 9, 'by': 10, 'jove': 1
1, 'my': 12, 'study': 13, 'of': 14, 'lexicography': 15, 'won': 16, 'prize': 17, 'this': 18, 'is': 19, 'short': 20, 's
entence': 21}

Sequence 1 in x
  Input:  The quick brown fox jumps over the lazy dog .
  Output: [1, 2, 4, 5, 6, 7, 1, 8, 9]
Sequence 2 in x
  Input:  By Jove , my quick study of lexicography won a prize .
  Output: [10, 11, 12, 2, 13, 14, 15, 16, 3, 17]
Sequence 3 in x
  Input:  This is a short sentence .
  Output: [18, 19, 3, 20, 21]
```

Building the Pipeline:

1. **Preprocessing:**

A. Load and Examine Data:

The inputs are sentences in English and the outputs are the corresponding translations in French.

B. Tokenization:

convert the text to numerical values. This allows the neural network to perform operations on the input data. For this project, each word and punctuation mark will be given a unique ID.

When we run the tokenizer, it creates a word index, which is then used to convert each sentence to a vector.

C. Padding:

When we feed our sequences of word IDs into the model, each sequence needs to be the same length. To achieve this, padding is added to any sequence that is shorter than the max length.

2. **Modeling:**

build, train, validation and test the model

Inputs: Each word is encoded as a unique integer that maps to the English dataset vocabulary.

Outputs: The outputs are returned as a sequence of integers which can then be mapped to the French dataset vocabulary

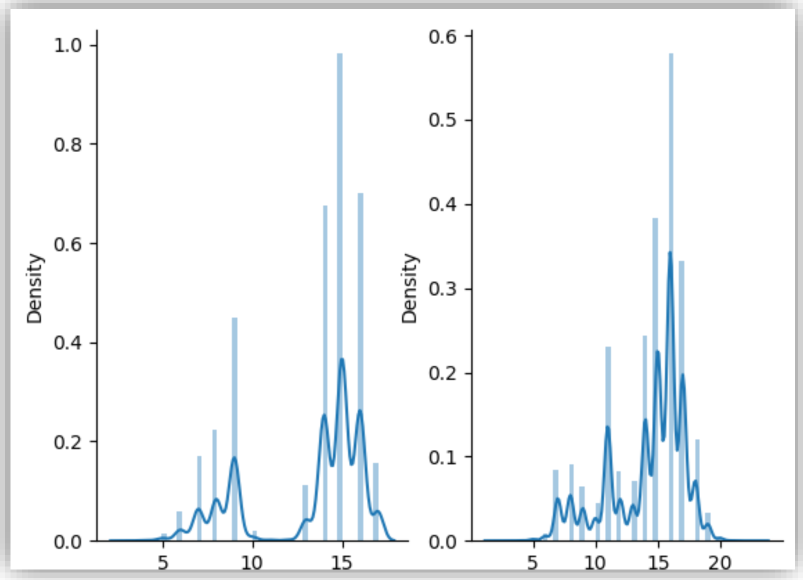
3. **Prediction:**

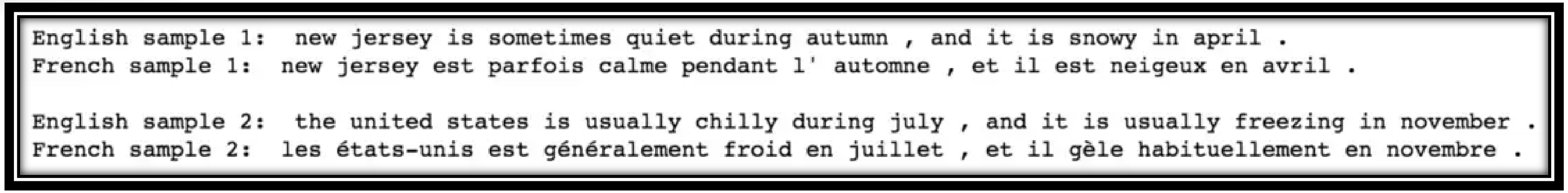
generate specific translations of English to French, and compare the output translations to the ground truth translations

- We use Keras for the frontend and TensorFlow for the backend in this project.

Dataset

- A data set is a collection of data
- The data contains sentences in English and French
- The data is divided into two files: sentences in English and sentences in English
- The amount of data is 1338863 sentences
- The data contains a ambiguity
- Average word count of the english text is about 11-16 words. Maximum reaching 17+.
- While that of the French text seemd to be around same 15-18 words. Maximum reaching around 21+.

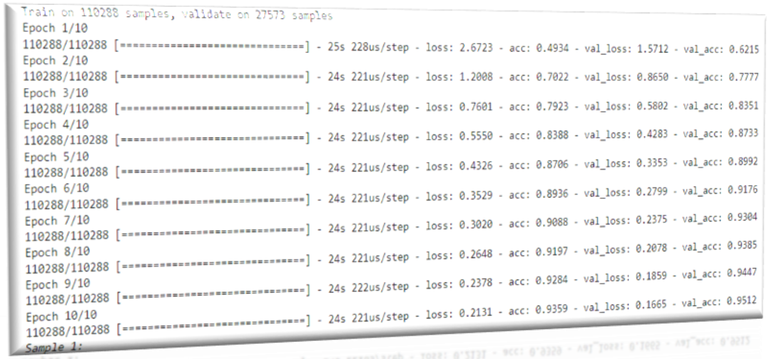




```
English sample 1:  new jersey is sometimes quiet during autumn , and it is snowy in april .
French sample 1:  new jersey est parfois calme pendant l' automne , et il est neigeux en avril .

English sample 2:  the united states is usually chilly during july , and it is usually freezing in november .
French sample 2:  les états-unis est généralement froid en juillet , et il gèle habituellement en novembre .
```

Current Approach



```
Train on 110288 samples, validate on 27573 samples
Epoch 1/10
110288/110288 [==============================] - 25s 228us/step - loss: 2.6723 - acc: 0.4934 - val_loss: 1.5712 - val_acc: 0.6215
Epoch 2/10
110288/110288 [==============================] - 24s 221us/step - loss: 1.2008 - acc: 0.7022 - val_loss: 0.8650 - val_acc: 0.7777
Epoch 3/10
110288/110288 [==============================] - 24s 221us/step - loss: 0.7601 - acc: 0.7923 - val_loss: 0.5802 - val_acc: 0.8351
Epoch 4/10
110288/110288 [==============================] - 24s 221us/step - loss: 0.5550 - acc: 0.8388 - val_loss: 0.4283 - val_acc: 0.8733
Epoch 5/10
110288/110288 [==============================] - 24s 221us/step - loss: 0.4326 - acc: 0.8706 - val_loss: 0.3353 - val_acc: 0.8992
Epoch 6/10
110288/110288 [==============================] - 24s 221us/step - loss: 0.3529 - acc: 0.8936 - val_loss: 0.2799 - val_acc: 0.9176
Epoch 7/10
110288/110288 [==============================] - 24s 221us/step - loss: 0.3020 - acc: 0.9088 - val_loss: 0.2375 - val_acc: 0.9304
Epoch 8/10
110288/110288 [==============================] - 24s 221us/step - loss: 0.2648 - acc: 0.9197 - val_loss: 0.2078 - val_acc: 0.9385
Epoch 9/10
110288/110288 [==============================] - 24s 222us/step - loss: 0.2378 - acc: 0.9284 - val_loss: 0.1859 - val_acc: 0.9447
Epoch 10/10
110288/110288 [==============================] - 24s 221us/step - loss: 0.2131 - acc: 0.9359 - val_loss: 0.1665 - val_acc: 0.9512
Sample 1:
```

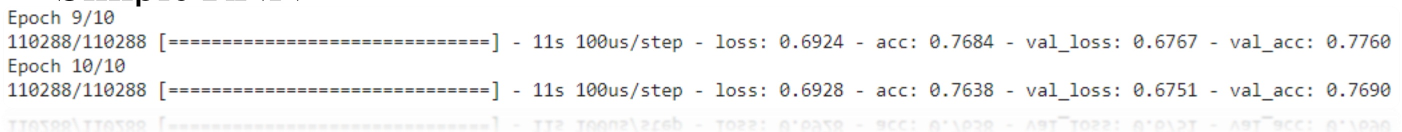
The original source repo for this project includes three main sections:

1. Preprocessing: tokenize and pad the dataset.

2. Models: where showcase three different network features on their own before combining them into the final model.

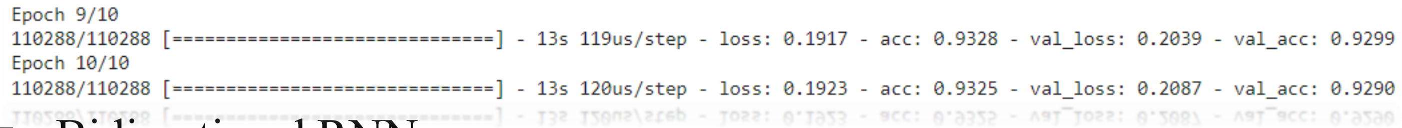
Here are the four architectures that will be shown in this section:

- Simple RNN



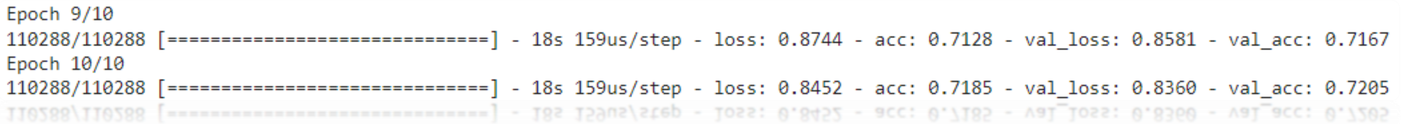
```
Epoch 9/10
110288/110288 [==============================] - 11s 100us/step - loss: 0.6924 - acc: 0.7684 - val_loss: 0.6767 - val_acc: 0.7760
Epoch 10/10
110288/110288 [==============================] - 11s 100us/step - loss: 0.6928 - acc: 0.7638 - val_loss: 0.6751 - val_acc: 0.7690
```

- RNN with Embedding



```
Epoch 9/10
110288/110288 [==============================] - 13s 119us/step - loss: 0.1917 - acc: 0.9328 - val_loss: 0.2039 - val_acc: 0.9299
Epoch 10/10
110288/110288 [==============================] - 13s 120us/step - loss: 0.1923 - acc: 0.9325 - val_loss: 0.2087 - val_acc: 0.9290
```

- Bidirectional RNN



```
Epoch 9/10
110288/110288 [==============================] - 18s 159us/step - loss: 0.8744 - acc: 0.7128 - val_loss: 0.8581 - val_acc: 0.7167
Epoch 10/10
110288/110288 [==============================] - 18s 159us/step - loss: 0.8452 - acc: 0.7185 - val_loss: 0.8360 - val_acc: 0.7205
```

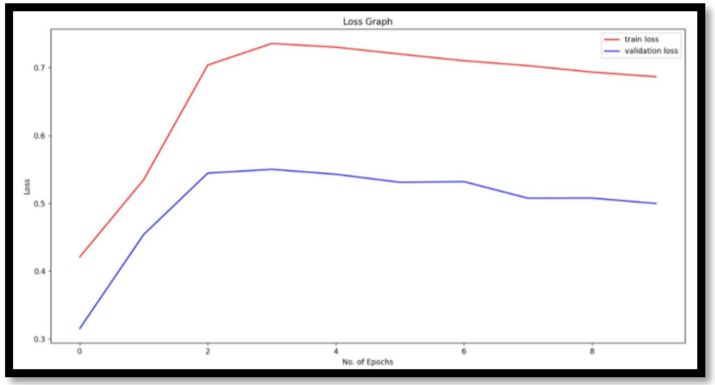
- Final Model

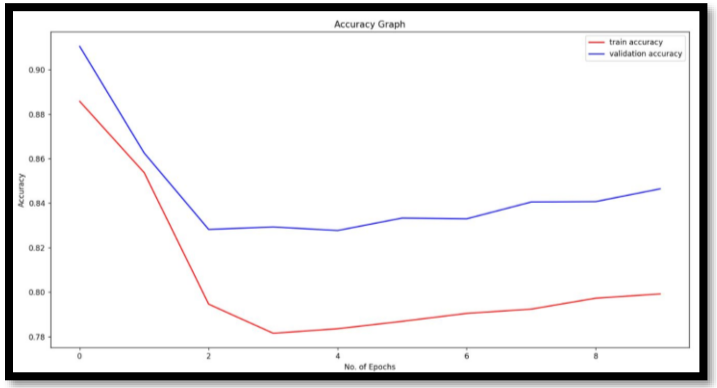
4. Prediction: where show how the trained model performs.

Results: 95% validation accuracy after only 10 epochs.

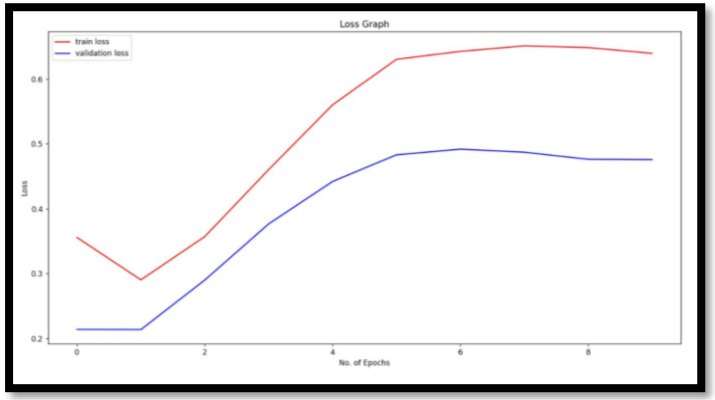
Before- Improvements - Accuracy

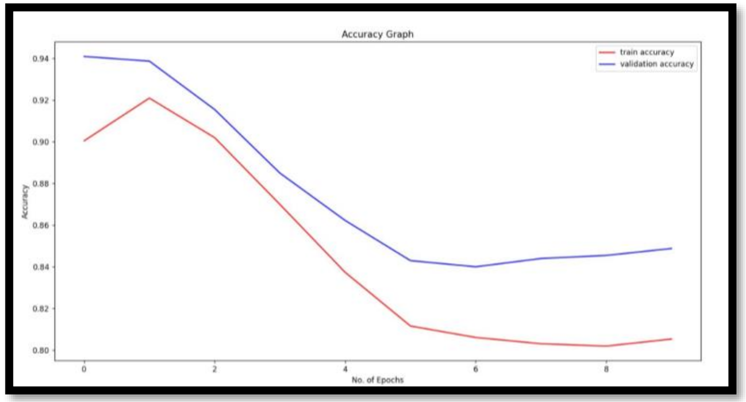
- **Train** on 1724 samples
- **Validation** on 20% samples
- **Dataset** on 130967 samples
- **Epochs** on 10

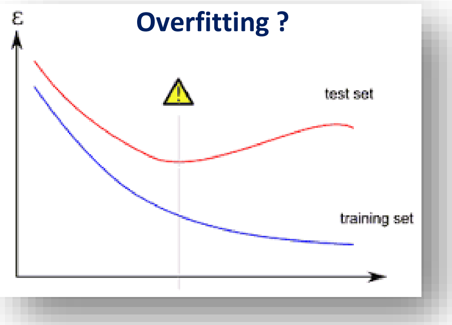




- **Train** on 1724 samples
- **Validation** on 20% samples
- **Dataset** on 62037 samples
- **Epochs** on 10







Improvements - Accuracy

optimizer	Size	Epochs	Results
Adamax	110288	19	good
RMSprop	110288	3	Not good
Adam	110288	10	Not good
Nadam	110288	3	Not good
Adadelta	110288	30	Not good
Adagrad	110288	20	Not good

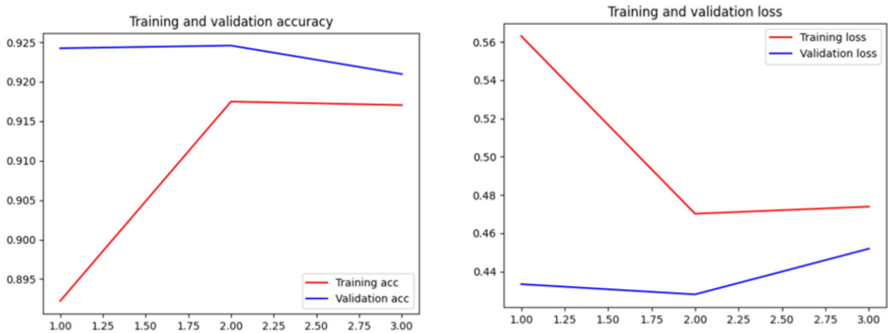
What is an optimizer?

Optimizers are algorithms or methods used to minimize an error function(loss function)or to maximize the efficiency of production. Optimizers are mathematical functions which are dependent on model’s learnable parameters i.e Weights and Biases.

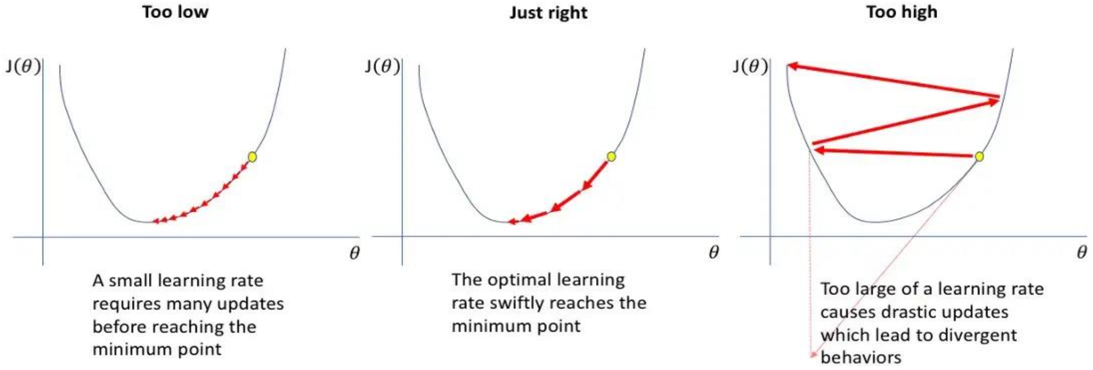
Adamax:

Adamax is a first-order gradient-based optimization method. Due to its capability of adjusting the learning rate based on data characteristics, it is suited to learn time-variant process, e.g., speech data with dynamically changed noise conditions.

RMSprop



Learning Rate



A small learning rate requires many updates before reaching the minimum point

The optimal learning rate swiftly reaches the minimum point

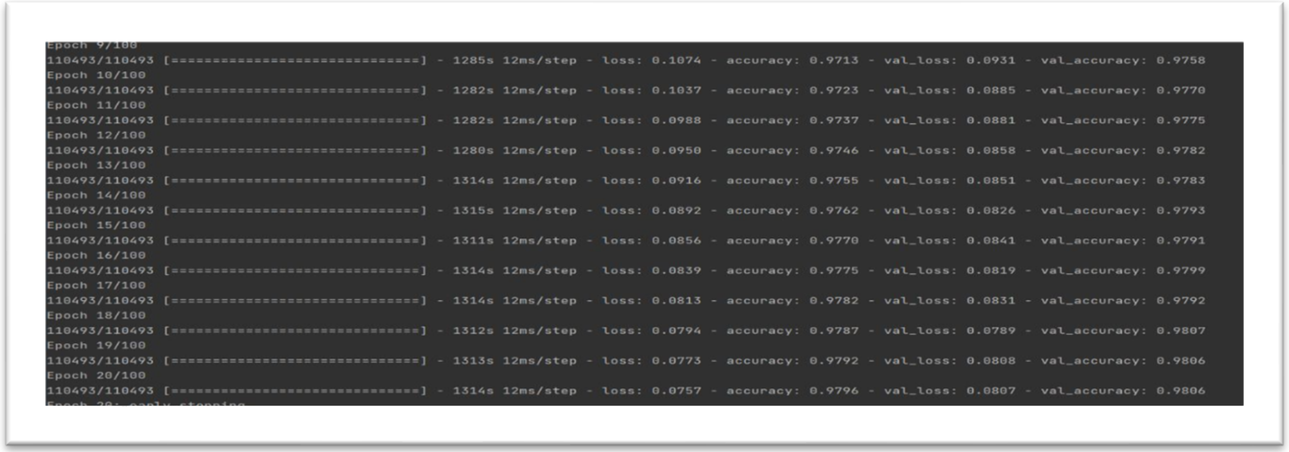
Too large of a learning rate causes drastic updates which lead to divergent behaviors

Improvements - Performance

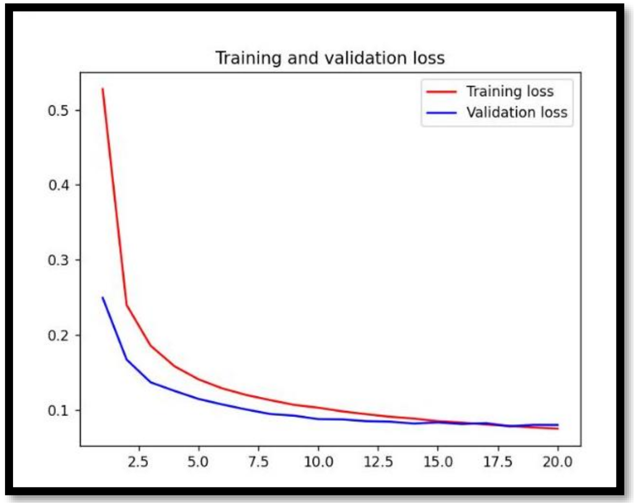
- Changing the optimization function – frome adam to adamax
- Add ambiguity – to data set and code
- Add a breakpoint - keras
- Check Point - The preprocessed data has been saved to disk
- Save data - pickle
- Graphical interface - tkinter
- Printing graphic results - matplotlib
- Printing the average word length in English and French - matplotlib
- Split dataset in train and validation and test data - sklearn
- Switch to Python - From jupyter notebook to Python

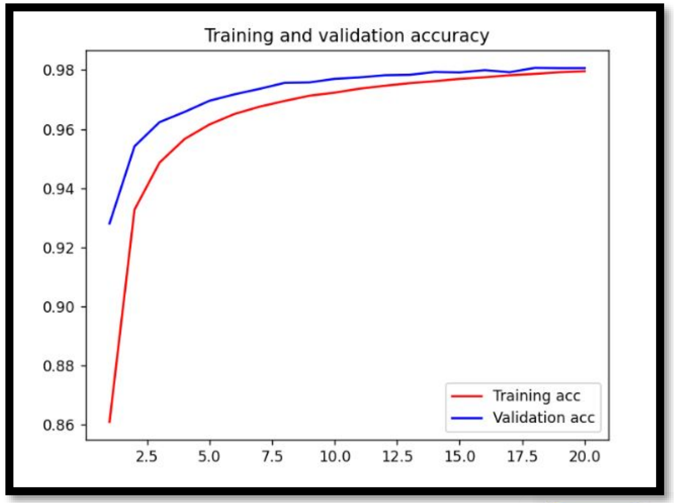
Results

- **Train** on 110493 samples
- **Validation** on 20% samples
- **Epochs** on 20
- **Accuracy**: 98%
- **Loss:** 7.5%



```
Epoch 9/100
110493/110493 [==============================] - 1285s 12ms/step - loss: 0.1074 - accuracy: 0.9713 - val_loss: 0.0931 - val_accuracy: 0.9758
Epoch 10/100
110493/110493 [==============================] - 1282s 12ms/step - loss: 0.1037 - accuracy: 0.9723 - val_loss: 0.0885 - val_accuracy: 0.9770
Epoch 11/100
110493/110493 [==============================] - 1282s 12ms/step - loss: 0.0988 - accuracy: 0.9737 - val_loss: 0.0881 - val_accuracy: 0.9775
Epoch 12/100
110493/110493 [==============================] - 1280s 12ms/step - loss: 0.0950 - accuracy: 0.9746 - val_loss: 0.0858 - val_accuracy: 0.9782
Epoch 13/100
110493/110493 [==============================] - 1314s 12ms/step - loss: 0.0916 - accuracy: 0.9755 - val_loss: 0.0851 - val_accuracy: 0.9783
Epoch 14/100
110493/110493 [==============================] - 1315s 12ms/step - loss: 0.0892 - accuracy: 0.9762 - val_loss: 0.0826 - val_accuracy: 0.9793
Epoch 15/100
110493/110493 [==============================] - 1311s 12ms/step - loss: 0.0856 - accuracy: 0.9770 - val_loss: 0.0841 - val_accuracy: 0.9791
Epoch 16/100
110493/110493 [==============================] - 1314s 12ms/step - loss: 0.0839 - accuracy: 0.9775 - val_loss: 0.0819 - val_accuracy: 0.9799
Epoch 17/100
110493/110493 [==============================] - 1314s 12ms/step - loss: 0.0813 - accuracy: 0.9782 - val_loss: 0.0831 - val_accuracy: 0.9792
Epoch 18/100
110493/110493 [==============================] - 1312s 12ms/step - loss: 0.0794 - accuracy: 0.9787 - val_loss: 0.0789 - val_accuracy: 0.9807
Epoch 19/100
110493/110493 [==============================] - 1313s 12ms/step - loss: 0.0773 - accuracy: 0.9792 - val_loss: 0.0808 - val_accuracy: 0.9806
Epoch 20/100
110493/110493 [==============================] - 1314s 12ms/step - loss: 0.0757 - accuracy: 0.9796 - val_loss: 0.0807 - val_accuracy: 0.9806
Epoch 20: early stopping
```

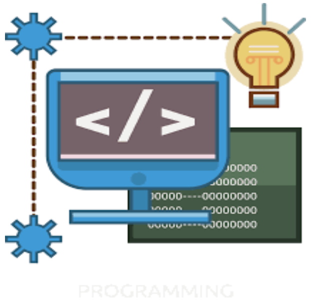




Code Questions

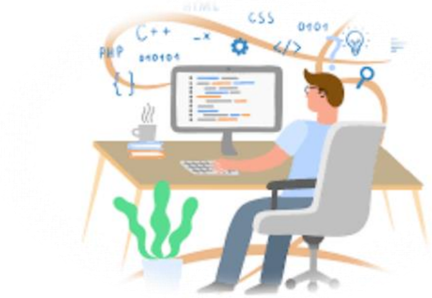
NIV KOTEK:

- Optimization
- Print the graph results
- Print average word length graph
- improve data
- graphic interface
- Ambiguity
- Check Point and Save data
- Switch to Python and divide the code into functions



SAEED ESAWI:

- Connection to cuda
- Running the models
- Optimization
- Print the graph results
- Stop training
- improve data
- Attempt to improve the model



Discussion – Open Questions & Future Work

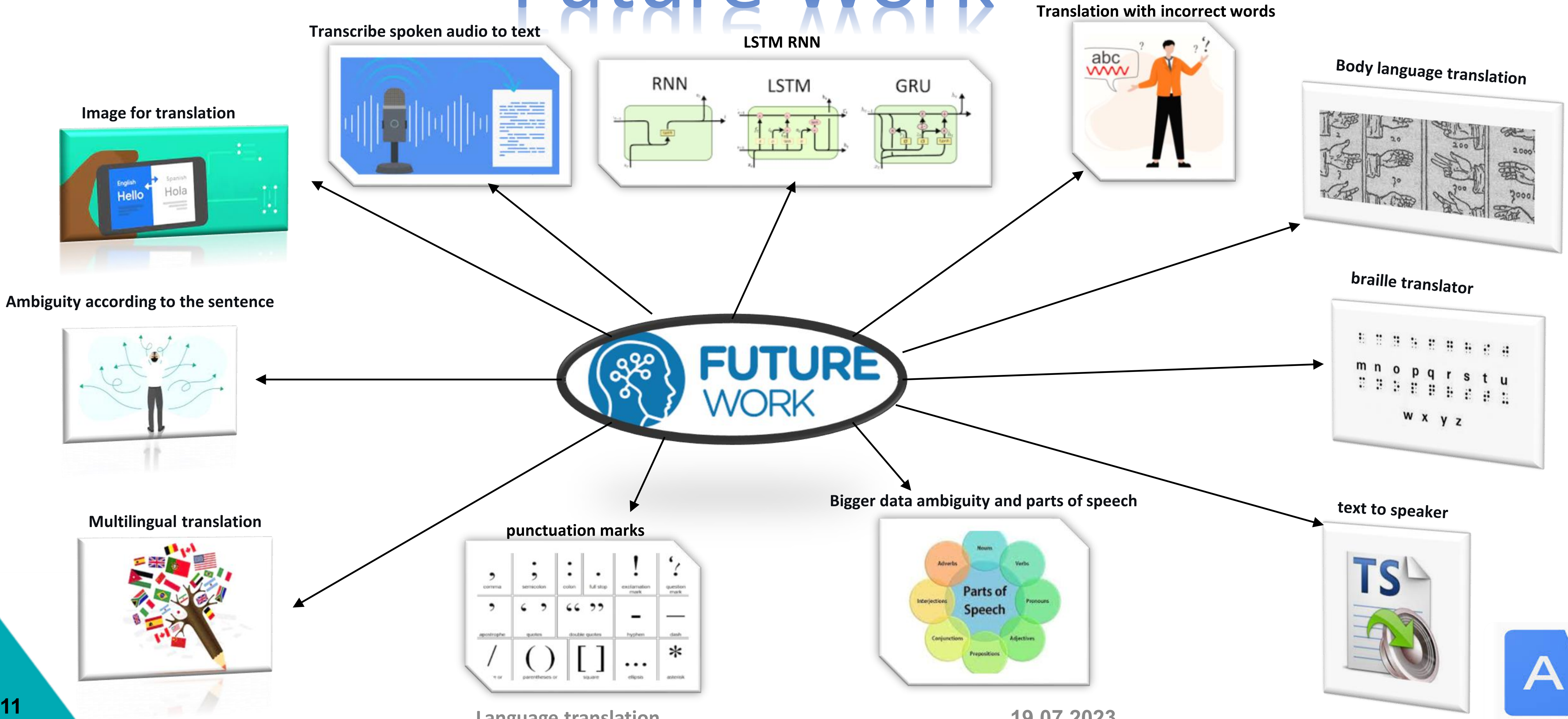


Image for translation

Transcribe spoken audio to text

LSTM RNN

Translation with incorrect words

Body language translation

braille translator

text to speaker

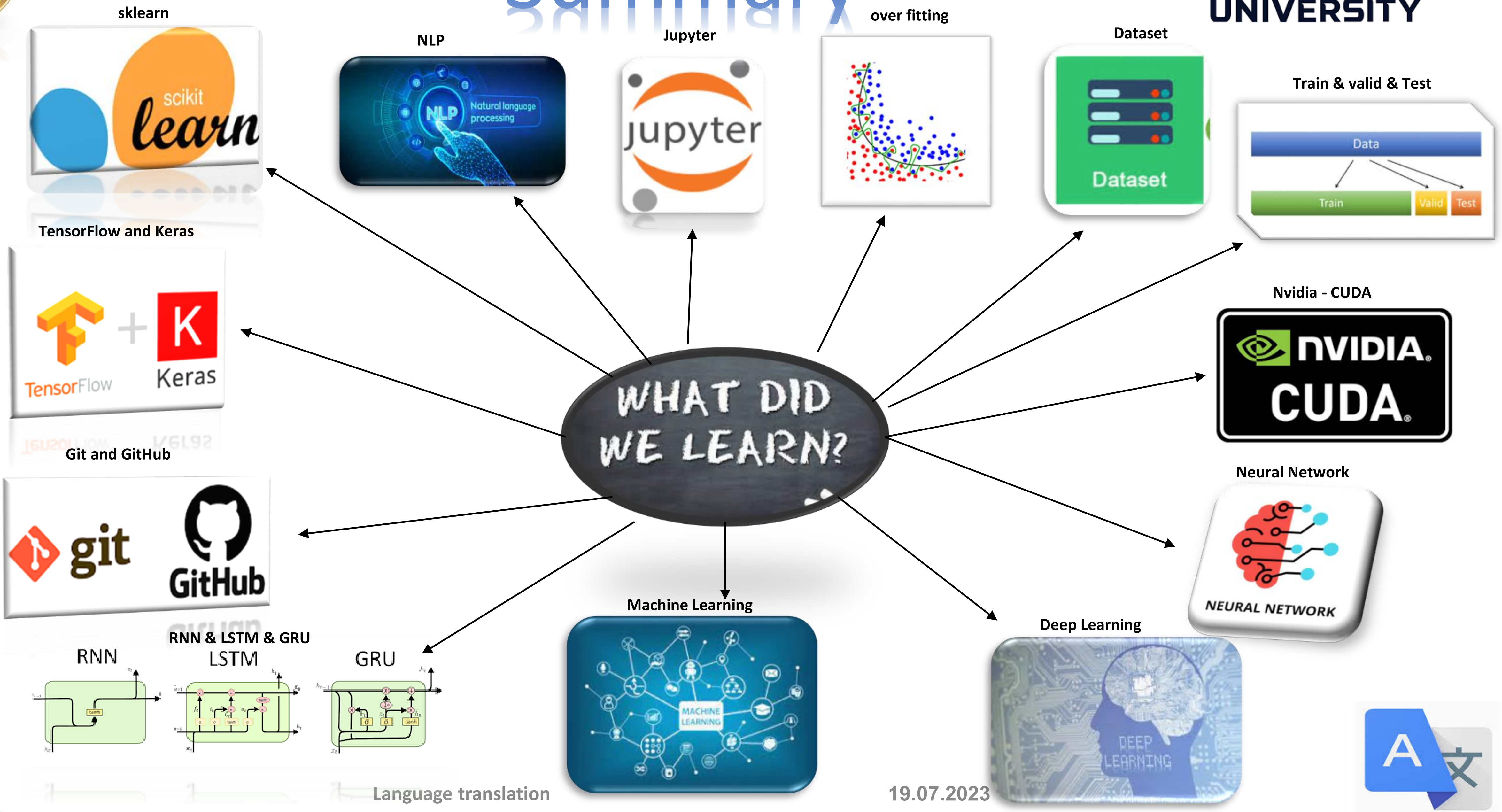
Bigger data ambiguity and parts of speech

punctuation marks

Multilingual translation

Ambiguity according to the sentence

Summary



WHAT DID WE LEARN?
sklearn
scikit learn
NLP
NLP Natural language processing
Jupyter
jupyter
over fitting
Dataset
Dataset
Train & valid & Test
Data
Train
Valid
Test
TensorFlow and Keras
TensorFlow
Keras
K
Nvidia - CUDA
NVIDIA
CUDA
Git and GitHub
git
GitHub
Neural Network
NEURAL NETWORK
RNN & LSTM & GRU
RNN
LSTM
GRU
tanh
Machine Learning
MACHINE LEARNING
Deep Learning
DEEP LEARNING
Language translation
A
文

QUESTIONS

ANY
QUESTIONS