**DATA SCIENCE CONSULTING Session 4**

February 20th 2023

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 0

Agenda

**1. Feedback on intermediary restitution**

2. Final presentation

3. Word Embeddings & Reccurent Neural Network

A. Word2Vec

B. FastText

C. RNN, Encoder Decoder, Sequence to Sequence

4. Language Model and Transformers

A. Attention mechanism and variants

B. Transformers

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 1

Feedback on intermediary restitution

**General comments** 

• Very satisfying level overall

• Good preparation work (good quality slides, research)

• Clear presentations’ structure

• Quality of delivery (good presentation skills)

**Areas of improvement**

• Some presentations lacked preliminary data analysis 

• Explicit the use of data research linked to the business issue and the client customer journey • Make sure you give enough details regarding your analysis and results • Do not forget to mention the limits of your analysis

• Do not forget best presentation practices (progress status, source, page number…)

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 2

Agenda

1. Feedback on intermediary restitution

**2. Final presentation**

3. Word Embeddings & Reccurent Neural Network

A. Word2Vec

B. FastText

C. RNN, Encoder Decoder, Sequence to Sequence

4. Language Model and Transformers

A. Attention mechanism and variants

B. Transformers

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 3

Final Restitution : 13/03

**Project team Client team** 

• Business director

• Data Science director

**Participants**

**Sponsor**

• Head of consumer gas & electricity (n-1 to Stéphane Michel, DG gas, renewables and power, exco member)

**Content**

**Objectives**

• Context

• Need

• Analysis scope

**Methodology**

• Data (what, how) • NLP methodologies used

**Analysis Results**

• Explain your conclusions, and the limits associated with them

**Recommendations**

• Based on the results of the analysis (including the

KPIs)

**Bonus – Next Steps**

• High level roadmap • Estimate the operational cost of implementation

**20mn presentation + 10mn questions Format**

Master X-HEC – 0602023 © Capgemini 2020. All rights reserved | 4

Workshop – How to build your storyboard?

Storyboarding helps you **structure your argumentation** and build the steps of an **effective communication**.

Design the scenario :

write the plan

Design the arguments

within the scenario

Produce views

supporting your

messages & add fluidity

Read it loud

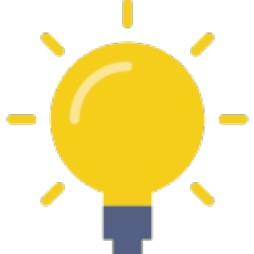
Repeat

RepeatRepeat

~~Master the content Manage transitions~~ Respect the time limit Write the introduction and the conclusion

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 5

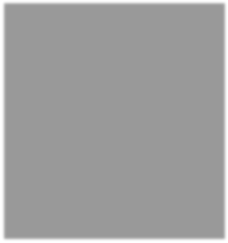
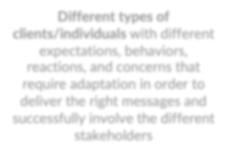
Brainstorm

**Start to storyboard your final presentation ! **

****15’

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 6

Adapting your messages to your audience **PROMOTING ENABLING ANALYSING CONTROLING**

****

**Different types of**

**clients/individuals** with different expectations, behaviors,

reactions, and concerns that require adaptation in order to deliver the right messages and

successfully involve the different stakeholders

Characteristics

What

he/she likes

What he/she doesn’t like

assertive, expresses feelings, has flair, stimulating,

enthusiastic, creative, talks a lot

To dream, to see the "big picture"

To be congratulated, see his/her work valued in public

To be restricted to tasks that are not very rewarding

Very detailed

presentations

kind, caring, friendly, cooperative, trustful, sympathetic, careful, sensitive, relaxed

To feel part of a group, to work as a team

To show consensus

To work on his/her own, with little

interaction with people

Conflict

patient, methodical, precise, low talker, thinker/writer,

accurate, measured, result-oriented,

thorough

Be provided with all the details

To feel secure

To depend too much on the work of

others

To be given unclear prospective subjects

determined, goal oriented, impatient, demanding, efficient, direct

Short, strait to the point presentations

ROI-focused

recommendations

Feeling wasting

his/her time

Inefficient/unclear presentations

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 7

Agenda

1. Feedback on intermediary restitution

2. Final presentation

**3. Word Embeddings & Reccurent Neural Network**

A. Word2Vec

B. FastText

C. RNN, Encoder Decoder, Sequence to Sequence

4. Language Model and Transformers

A. Attention mechanism and variants

B. BERT

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 8

Data pipeline



**Data Cleaning**

**Data Collection**

**Topic Extraction**

**Word Embedding Sentiment Analysis**

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 9

Word embedding reminder

**Definition: Word embedding is a type of word representation that allows words with similar meaning to have a similar representation.[1]**

Machine Learning and Deep Learning algorithms can only take numeric input. This means that for NLP we’ll need to transform our text data (words, tweets, full documents…) into numerical (vectors) by using embedding techniques.

**Some embedding techniques:**

• Traditional approaches:

• Latent Semantic Indexing (LSI) • Word2Vec

• FastText

• Advanced techniques: Transformers • GPT-3

• BERT

• CTRL

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 10 [1] GOLDBERG, Yoav. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 2017, vol. 10, no 1, p. 1-309.

Agenda

1. Feedback on intermediary restitution

2. Final presentation

**3. Word Embeddings & Reccurent Neural Network**

**A. Word2Vec**

B. FastText

C. RNN, Encoder Decoder, Sequence to Sequence

4. Language Model and Transformers

A. Attention mechanism and variants

B. Transformers

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 11

Word2Vec

• Word2Vec is an embedding method that which leverages the context of target words.

• It uses the surrounding words to represent target words with a Neural Network whose hidden layer encodes the word representation • It provides a representation of each of the V words in the vocabulary by mapping context and targets in a sentence. • Two different methods are available:

**target**

**word context context target**

**CBOW**

Takes the **context** of each word as the input and tries to predict the word corresponding to

the context being the **target**

**word**

**Skipgram**

Takes the **target** word and tries to predict the **context** words of that target word and 

produce

representations

*Corpus example*: “**The cat jumped over the fence**” {“The”, “cat”, “over”, “the”, “fence”} => **context words** {“jumped” } => **target word** 

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 12

Skipgram Architecture

***V*** : Number of words in the Vocabulary ( Size of vocabulary*) (V=100 for example)* ***N*** *:* Dimensions of each embedding

One sentence composed of words from Vocabulary “**The cat jumped over the fence**”

� ���

��**: Target word**

� ���

� ���

� ���

� **Context words :** ���

� ��)

� ���

We want to maximize the log likelihood

**Matrix** �!

V x N

**Matrix** �"

N x V

Softmax

� ��)

� ���

� ��)

� ���

� ��)

� ���

� ��)

� ���

∑!∈#- log � �! | �$

**On T words**

%

& ∑!∈#- log � �! | �$

**One hot encoding of Target word**

� � � = ��� � �

**Embedding of Target word**

**N** � �

(� �(, �) )\*" ∈, **V** � �

& ∑$'%

**V** � �

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 13

Zoom on softmax function

• Each word �, have 2 representaRons : �# (when w is a target word) and �# (when w is a context word) such that : & �#.

� �$, �% = < �#- , �#. > = �#-

• s is a scoring function between a word �$ and a context word �%

**0**

**0**

**1**

**0**

**0**

**0**

**0**

**0**

**One hot encoding of Target word**

**Matrix**

�!

V x N

**Embedding of Target**

**word**

**Matrix** �"

N x V

**2 3 1 2 5 4 3 2**

Softmax �'0

\* �~~'~~/

∑()!

**0.028 0.075 0.01 0.028 0.55 0.204 0.075 0.028**

� �! �$) =�( )-,). , �( )- ,)/

∑+'%

� � � = ��� � �

**N** � �

**V** � � (� �(, �) )\*" ∈, **V** � �

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 14

Negative Sampling Skipgram

**Problems with previous softmax formulation :**

1. Implies to predict only one context word �# given a word �!. ❌

2. Computing is very expensive ❌

For example : if V = 10 000 words and N = 300, the matrix will have size = 3M. For each training sample, we will need to update 3M weights.

If we have 1 billion training samples, we will need to update 3M weights 1 billion times. For a word �! :

**Alternatives :**

1. Predict context words as a set of independant binary classification (presence or absence of a word in context) ✅ 

2. Limit the numbers of weights that will be updated by each training sample ✅

3. Performing subsampling ✅

• New training dataset formulation : �!, �" ���ℎ � ����� � , � = 1 �� �" �� �� �ℎ� ������� �� �! and 0 if not

• We consider all context words as positive examples and sample negatives at random from dictionary.

• For a chosen word context �#, using the binary logistic loss, we obtain the following negative log-likelihood, where �!,# is a set of negative examples sampled from the vocabulary :

log 1 + �%& '#, '" + 8 ( ∈ �#,"

log 1 + �& '#, (

Binary loss for true context Binary losses for negative samples

• By denoting the logistic loss function ℓ ∶ � ⟼ log 1 + �%+ , we can re-write the objective as :

,

- [ ∑#∈/# ℓ � �!, �# + ∑(∈�#," ℓ −� �!, � ]

- ∑!.,

*Sources : http://mccormickml.com/2017/01/11/word2vec-tutorial-part-2-negative-sampling/*

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 15

Negative Sampling Skipgram

+

, [ ∑/∈1, ℓ � �-, �/ + ∑2∈�,,-ℓ −� �-, � ]

, ∑-.+

�� : set of context words of word �$ ��,� : set of negative samples of word �$

Binary loss for true context Binary losses for negative samples

context word center/target word

sliding window

Machine learning (ML) is the study of **computer algorithms that** improve automatically through experience. It is seen as a part of artificiel intelligence. Machine learning algorithms build a model based on sample data, known

as « training data », in order to make predictions or decisions without being explicitly programmed to do so. negative samples

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 16

Machine learning is the study of **computer** algorithms that improve automatically through experience

(computer, algorithms)

(Machine, Learning)

(study, of)

(6,7) … (through, experience) (improve, automatically)

(8,9)

(computer, algorithms) (8,9)

(1,2)

(computer, algorithms) (8,9)

1

(computer, learning) (8,2)

(computer, learning) (8,2)

0

(10,11)

(computer, improve) (8,10)

(computer, improve) (8,9)

0

(computer, data) (8,23)

(computer, data) (8,23)

0

(13,14)

(computer, request) (8,34)

(computer, request) (8,34)

0

Word Context Words Labels

8

9 2 10 23 34 1 0 0 0 0

8

2 11 28 46 22 1 0 0 0 0

**.**

**.**

**.**

8

39 27 4 29 10 1 0 0 0 0

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 17

Agenda

1. Feedback on intermediary restitution

2. Final presentation

**3. Word Embeddings & Reccurent Neural Network**

A. Word2Vec

**B. FastText**

C. RNN, Encoder Decoder, Sequence to Sequence

4. Language Model and Transformers

A. Attention mechanism and variants

B. Transformers

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 18

FastText

**Reminder about word2vec limitations :**

• Ignores the internal structure of words by assigning a distinct vector to each word ❌ • Represents a limitation for rich languages with large vocabularies and many rare words or misspellings ❌

**FastText introduces *Subword Information* :**

• Learning word representations from **character** �**-grams** as the **sum of character** �**-gram** vectors using a different scoring function �. • For instance, for the word �ℎ��� with � = 3 :

< �ℎ, �ℎ�, ℎ��, ���, �� >

Consider a dictionary of �-grams of size � and given a word �, we denote �# ⊂ {1, … , �} the set of �-grams appearing in �. We associate a vector �/ to each �-gram � and represent � word by the sum of the vector representations of its �-grams. • To learn these �-grams vector representations, we use the following scoring function :

�# = Σ/ ∈ �0 �/

� �, � = �)- �! = (Σ/ ∈ �0 �/)⊤ �% = Σ. ∈ �1�.- �!

where � is the context word.

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 19

FastText

• Words �!, … , �2 and label � (classification)

• **bag of n-gram features** to capture partial words order information

• **Weight matrix** � embedding each n-gram features �! … �3

• **Words representations averaging** to get the hidden text representation

• Linear classifier : **weight matrix** � for linear projection then **softmax-like function** to get probability distribution Different from word2vec

�! �"

�#

.

.

.

�$

Separate each word in

n-grams

�! �"

�#

.

.

.

�%

����ℎ� ������ �

Embedding of �2

Embedding of �1

Embedding of �3

.

.

.

Embedding of �4

����ℎ� ������ �

���0

�: ������� − ����

��������

�(���5) : Probability

**Words of document d**

�� **: n-grams of document d**

**A**�� **: Embedding of n-grams**

**Probability distribution**

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 20

Softmax Approximation : Hierarchical Softmax

• Hierarchical softmax uses a binary tree to represent all words in the vocabulary. Each leaf of the tree is a word, and there is a unique path from root to leaf.

• In this model, the probability of a word *w* given a vector *wi*, *P*(*w*|*wi*), is equal to the

probability of a random walk starting in the root and ending in the leaf node

corresponding to *w*.

• The main advantage in computing the probability this way is that the cost is

only *O*(log2(*V*)), corresponding to the length of the path.

• Let � �,� be the �-th node on the path from the root to � and � � the length of

the path, so �(�, 1) = root and �(�, � � ) = �.

• For any inner node �, let �ℎ(�) be the left child of � and [�] = 1 if � is true and −1

otherwise. Thus, hierarchical softmax defines �(�|�5) as follows :

6 # 7!

� � �5 = [

()!

where � � = !

�( � �,� + 1 = �ℎ � �,� . �8 #,( 9 :�#6)

!;<=>(7@)and which sums to 1.

Source : https://flavien-vidal.medium.com/

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 21

To retain from Word2Vec and Fastext

Two embedding approaches that have one thing in common, they have exactly one embedding for one word or n-gram of a word. They do not look at the contextual relationship of a specific word, or only in a very small range and limited way (skipgram model).

**Word2Vec FastText**

• Provides a representation for each word in

the vocabulary.

• Limited when dealing with misspellings and

complex languages.

• Does not consider morphology of words.

Two main problems that present these methods are**:**

• Representation of every n-gram. • Size of the dictionary can cause a high memory requirement.

• Always the same representation for a word type, regardless of the context in which a word token occurs. • Only one representation for each word, but each one has different aspects, including semantics, syntatic behaviour, and register/connotations. 

In order to fix them, we should consider some more advanced techniques that will use the overall context of a word in a given sentence. A same word will have different representation depending on the sentence they are in. Some examples of these methods are:

• ELMO (Embeddings from Language Models).

• Transformers techniques such as GPT-3, BERT or CTRL.

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 22

Hands-on 1

45’

**Skipgram with negative sampling using TensorFlow **17: 20



Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 23

Break

20’

**See you after 20 min!**

****

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 24

Restitution

5’

**Could you implement Word2Vec ?**

****

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 25

Agenda

1. Feedback on intermediary restitution

2. Final presentation

**3. Word Embeddings & Reccurent Neural Network**

A. Word2Vec

B. FastText

**C. RNN, Encoder Decoder, Sequence to Sequence**

4. Language Model and Transformers

A. Attention mechanism and variants

B. Transformers

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 26

Reccurent Neural Networks (RNN)

• Consider a variable-length sequence x = �!, … , �:& such that �$ ∈ ℝ2 and representing the sequence of words �!, … , �: • A simple recurrent neural network encodes such sequences at each step � in the following way :

ℎ#

ℎ@ = � ℎ@AB, �@ = tanh �� ℎ@AB + �� �@

where :

• �$ is the word embedding of the word �$ • �� ∈ ℝC**×**C**,** �� ∈ ℝC**×**2**,** ℎ$ ∈ ℝC • � is called the RNN Cell

• **h** = ℎ!, … , ℎ:& are called hidden states

ℎ#$%

�&

�' RNN CELL

• More sophisticated RNN cells : **Long-Short Term Memory** (LSTM), **Gated Recurrent Units** (GRU) �#

• Simple RNNs are struggling to capture long term dependency from the input sequence

ℎ! ℎ)\*!ℎ)

ℎ(RNN CELL

�!

RNN CELL

�"

… RNN CELL

�)

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 27

Reccurent Neural Networks (RNN)

**Principles**

• Handle variable-length sequence

• Preserve word order

• Capture sequential information

• Encode sequence/sentence information in continuous vectors **Bidirectional Recurrent Neural Networks**

ℎ% ℎ1 ℎ2 ℎ& ℎ% ℎ1 ℎ2 ℎ&

• Concatenation of forward and backward RNN’s hidden states to build word representation containing the summaries of both the preceding words and the following words

• The forward RNN �⃗ reads the input sequence from �! to �: and builds the forward hidden states ℎ!, … , ℎ:

• The backward RNN �⃖ reads the reverse input sequence from �: to �! and builds the backward hidden states ℎ!, … , ℎ: • Finally, get ℎ!, … , ℎ: such that ℎ( = ℎ( ; ℎ(

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 28

Reccurent Neural Networks (RNN)

**Use RNN to learn a probability distribution over a sequence (e.g. language modeling) :**

• Train the network to predict the next symbol in the sequence � �$ | �$7!, … , �! . For � ∈ {1, … , �}, � �@,E | �@AB, … , �B = exp wE h@

H exp wEF h@

∑EFGB

where w( are the rows of the weight matrix **W** ∈ ℝE×C

softmax(W .)

�' RNN

ℎ#

• By combining these probabilities, we can compute the probability of the **sequence x** : I

ℎ#$%

CELL

� x = / @GB

� �@ �@AB, … , �B)

�&

**Use RNN for sequence classification (e.g. sentence classification)** �#

ℎ3

�%

ℎ% ℎ&4% ℎ& Positive sentiment …

Negative sentiment

�1 �&

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 29

Long Short Term Memory (LSTM)

**Type of RNN as a solution to preserve information over many time steps (Problem of Vanishing gradients)** 

On step �, there is a hidden state ℎ, and a cell state �: Both are vectors length �. The LSTM can erase, write and read information from the cell

**The selection of which infromation is erased/written/reas is controlled by three corresponding gates** 

✔ The gates are also vectors length n

✔ On each timestep, each element of the gates can be open (1), closed (0), or somewhere in between

�� = � ����&� + ���� + �� **Forget gate**: controls what should be kept or forgotten from previous cell state

�� = � ����&� + ���� + ��

**Input gate**: controls what part of the new cell content are written to cell

**LSTM Cell**

Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

�� = � ����&� + ���� + �� **Output gate**: controls what parts of cell are output to hidden state �P� = **tanh** ����&� + ���� + �� **New cell content** :new content to be written to the cell �� = �� ∘ ��&� + �� ∘ �P�**Cell state**: forget some content from last cell state, and input some new cell content 

�� = �� ∘ ���� �� **Hidden state**: output some content from the cell

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 30

Encoder-Decoder Architecture

**Output of decoder => **Using the hidden state at the current time step together with the respective weight W(S)

Tapez une équation ici.

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 31

Encoder-Decoder Architecture

**I. Encoder**:

• Processes the sequence and then returns at the end an encoding vector of the whole series • Several LSTM cells, for each cell it accepts a single element of the input sequence and propagates forward

• Generates a context vector that summarizes the hidden layers and encapsulates all the information **II. Decoder**: 

• The context vector is then taken as the input to the decoder in order to make the predictions. • Each recurrent unit accepts a hidden state from the previous unit and produces and output as well as its own hidden state.

�

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 32

Encoder-Decoder Architecture

**Main Drawbacks of the Encoder-Decoder Model:**

✔ **Architecture with Limited memory**: The final state is where all the information of the sentence is -> the more the length is fixed of the W matrix, the lossier the neural network will be

✔ **Deeper neural network => hard to train** i.e. for RNN, the length of the neural network is dependant on the sequence’s length

=> **Problem of** Vanishing Gradients: gradients that are used to update the weights shrink For more Robust and lengthy sentences > **Attention Models**

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 33

Sequence to Sequence Models

Use Cases of the **Sequence to Sequence** Models (seq2seq):

• Google Translate

• Chatbots

• The applications are related to any sequence-based problem when the input and outputs have different sizes (Machine Translation and Speech Recognition)

• Most Common architecture to build Seq2Seq models is Encoder-Decoder Architecture



Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 34

Agenda

1. Feedback on intermediary restitution

2. Final presentation

3. Word Embeddings & Reccurent Neural Network

A. Word2Vec

B. FastText

C. RNN, Encoder Decoder, Sequence to Sequence

**4. Language Model and Transformers**

**A. Attention mechanism and variants**

B. Transformers

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 35

Attention Model

The fixed size context-vector bottleneck was one of the main motivations by **Bahdanau et al. 2015**, which proposed a similar architecture but with a crucial improvement**:**

✔ **Encoder**: Bi-Directionnal RNN with a forward and backward hidden states

=> Concatenation of the two hidden states= encoder state at a given position

***Why?*** Include both preceding and following words in the representation of an input word

✔ **Decoder:** Search mechanism at the decoder level => Look at the whole input sentence

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 36

Attention Mechanism

In the RNN Decoder, attention mechanism allows to build for each time step

� a more relevant context vector �$ 

� �$ | �!, … , �$7! , x = � �$7!, �$, �$

where �U is the RNN hidden state for time �, computed by :

�@ = �JKLMNKO �@AB, �@AB, �@

**Attention scores, can be computed with several variants, here this is**

**a simple dot product between each hidden state** �� **and decoder**

**hidden state at** �**,** ��

�$ = �$:ℎ!, … , �$:ℎ2

Unlike previous approach, the probability is conditioned on a distinct

context vector �U for each target word �U

�U depends on the RNN encoder hidden states sequence ℎ!, … , ℎ: and is

weighted according to each encoded input vector :

�@ = softmax(�@)

Q �@PℎP

�P = ΣP G B

The context vector is then concatenated to the decoder hidden state to

produce the prediction at time step t

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 37

Attention Mechanism

Seq2seq with fixed-length Seq2seq with attention **https://zhuanlan.zhihu.com/p/37290775 https://zhuanlan.zhihu.com/p/37290775**

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 38

Attention Mechanism is everywhere

• Significantly **improves performances** 

• Solves the bottleneck problem

• allows decoder to **look directly at source**; bypass bottleneck

• helps with vanishing gradient problem

• **Provides some interpretability**

• Attention is a general Deep Learning technique that can be leverage for a lot of tasks • Machine Translation

• Sentence classification • Sequence tagging • Question Answering • Image captioning • …

Source: Neural Machine Translation By Jointly Learning To Align And Translate

Source: Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 39

Transformers

▪ Transformer: stack of 6 layers 

▪ 6 layer encoder stack on the left

▪ 6 layer decoder stack on the right

▪ No RNN, LSTM et CNN

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 40

Transformers

Positional encoding added to word embeddings 

• Provide sine and cosine functions for the

positional encoding for each dimension I of the

word embedding vector



• Add positional vector to embedding vector

before fed to the first encoder

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 41

Transformers

The **encoder** consists of the following 

components:

• Input embeddings ( including the positional

embedding => multi-head attention

• Add word embeddings to the output of the

attention and normalize

• After normalizations, feed forward neural 

network

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 42

Transformers

**Decoder** consists of the following components: 

• Output embeddings Y => fed into multi-head

attention

• Information is passed to another multi-head

attention (every position in the decoder to see

all the positions in the input sequence)

• Result is passed through a feed forward neural network

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 43

Transformers

**Classifier:**

• Last step of the transformers model

• Linear transformation and softmax to convert

decoder output to predicted next token

probabilities

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 44

Summary of the session

**Learning word embeddings**

• Negative Sampling and hierarchical softmax are two methods to tackle the computation complexity of Softmax function • Word embeddings can be pre-trained and use as initialization weights for any NLP tasks

• FastText can create infer embeddings of words that are not in the training set, thanks to character level representations • Both Word2vec and FastText fail at effectively capturing polysemy

**Learning contextual representations of text**

• RNN is a family of flexible architectures to model variable length sequences, hence are efficient for NLP problems • LSTMs (and GRUs) are powerful at capturing long term dependencies compares to simple RNN • Using bidirectional RNN is preferable (when applicable) to capture the context at each time step • Seq2seq architecture is particulary suited for Machine Translation or any NLG tasks

• Attention mechanism is a general deep learning technique that helps the model to focus on particular parts of the input • Attention weights can be used to provide some interpretability

• The transformer has had great success in NLP. Many pre-trained models such as GPT-2, GPT-3, BERT, XLNet, and RoBERTa demonstrate the ability of transformers to perform a wide variety of NLP-related tasks such as machine translation, document summarization, document generation, named entity recognition, and video understanding.

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 45

**Word Embeddings**

References

• Efficient Estimation of Word Representations in Vector Space (2013) 

• Distributed Representations of Words and Phrases and their Compositionality (2013) • Bag of Tricks for Efficient Text Classification (2016)

• Enriching Word Vectors with Subword Information (2017)

• Advances in Pre-Training Distributed Word Representations (2017)

• Misspelling Oblivious Word Embeddings (2019)

**Embeddings, Sequence-to-Sequence, Neural Machine Translation & Language Models** • Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation (2014) • Sequence to Sequence Learning with Neural Networks (2014) 

• Neural Machine Translation by Jointly Learning to Align and Translate (2014)

• Effective Approaches to Attention-based Neural Machine Translation (2015)

• A Structured Self-attentive Sentence Embedding (2017)

• Transformer: Attention is All You Need (2017)

• Deep Contextualized Word Representations (2018) (**ELMo**)

• **BERT** : Pre-training of Deep Bidirectional Transformers for Language Understanding (2018) • GPT-2: Language Models Unsupervised Multitask learners (2019)

• **GPT-3** : Language Models are Few-Shot Learners (2020)

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 47

Course evaluation

**Did you like that course ? It’s time to share your feedbacks !**

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 48

**Thank you for your attention GOODBYE !**

February 20th, 2023

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 49

Hierarchical Attention Network (HAN) for Document Classification

*Hierarchical structure that mirrors the hierarchical structure of documents; (ii) it has* 

*two levels of attention mechanisms applied at the wordand sentence-level, enabling*

*it to attend differentially to more and less important content when constructing the* 

*document representation.* ���� = − ∑ log �23

The document vector � is a high level representation of the document and can be

� = softmax(�4� + �4) 

�+ = tanh �1ℎ+ + �1 �+ = �/!#/% 

∑+ �/!#/%

used as features for document classification 

We use the negative log likelihood of the correct labels as training loss where � is the label of document �.

�� the sentence attention vector 

� is the document vector that summarizes all the information of sentences in a

� = n+

�+ℎ+

document.

ℎ+ = ��� �+ , � ∈ [L, 1] ℎ+ = ���(�0),� ∈ [L, 1] ℎ+ = [ℎ+; ℎ+] 

�+, = tanh �.ℎ+, + �. �+, = �/!"# /$ 

∑, �/!"# /$

�+ = n+�+,ℎ+, 

�+, = �-, �+, ,� ∈ 1, � , ℎ+, = ��� �+, ,� ∈ 1, � , ℎ+, = ��� �+, ,� ∈ 1, � . ℎ+, = [ℎ+, ; ℎ+, ]

ℎ+ summarizes the neighbor sentences around sentence �� but still focus on i-th sentence. 

introducing attention mechanism to extract words that are important to the meaning of the sentence and aggregate the representation of those informative words to form a sentence vector �� 

bidirectional GRU to get annotations of words by summarizing information from both directions for words at time step � of the sentence �, and therefore incorporate the contextual information in the annotation. 

Master X-HEC – 20022023 © Capgemini 2023. All rights reserved | 50