POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Politecnico
di Torino

Master's Degree Thesis

Video lectures summarization

Supervisors

Prof. Laura FARINETTI

Prof. Luca CAGLIERO

Dott. Lorenzo CANALE

Candidate

Irene BENEDETTO

July 2021

Abstract

With the recent advancements in e-learning platforms and the spread of distance learning, there is a growth in interest in generating content accompanying traditional video lessons. This thesis work aims to compare different techniques that derive from the NLP and deep learning fields to summarize the content of video lecture transcripts. Text summarization and related tasks have been extensively studied in the literature; conversely, transcript summarization has not been fully explored. Unlike the former, the transcripts summarization task presents some more critical issues: typically raw transcripts do not contain punctuation marks and are difficult to segment and process for traditional text summarization techniques. Moreover, dialogues differ from plain texts in some syntactical features. Lastly, it is important to point out that the conversion from audio to text may introduce some errors due to the translation process. These limitations complicate the usage of text summarization models in the case of speech transcripts. To the best of our knowledge, at the moment, there is no study focusing on summarization of educational content, given the absence of a specific dataset. For this reason, the thesis focuses on analyzing model domain adaptation capabilities and presents a novel and ad-hoc dataset, based on MIT OpenCourseWare video lectures. In the first part, the work explores different approaches for punctuation restoration for speech transcripts. In the second part, the dissertation examines the state-of-the-art approaches in text summarization and in particular, summarization in the meeting domain, the closest domain available in the literature with respect to the educational one. The thesis dedicates a section for evaluation of the results on an educational dataset EduSum, containing transcriptions and summary of MIT video lectures and Politecnico's video lectures. We discovered that, although state-of-the-art meeting summarization models are pre-trained on large datasets and finetuned with meeting data, they have poor performance and generalization capability when the domain changes. In the thesis a novel approach is proposed, that has proven to be more robust to domain shift, even if it uses simpler and lightweight models.

Acknowledgements

Thanks to all the supervisors: I thank Professor Laura Farinetti, Professor Luca Cagliero and Doctor Lorenzo Canale for all the help given to me in the realization of this thesis work. Thanks for the continuous dialogue, which, despite the distance, has always been of great teaching.

My most heartfelt thanks go to my parents, I would like to thank you for trusting me over the years, in all the choices that have brought me up to here. This thesis is also due to your incessant support.

A thought goes to all the members of my family, for having always been close to me. In particular, to my grandmother, who is probably the person that would most appreciate this work unconditionally.

A huge thanks goes to a person who had the patience to teach me and showing me day-to-day the concrete meaning of the term *resilience*.

Finally, I would also like to acknowledge my friends, colleagues, and all who have faced with me these years. Thanks for every synergistic exchange of ideas and for always being of encouragement.

Irene

Table of Contents

List of Tables

List of Figures

Acronyms

AI

Artificial intelligence

AIEd

Artificial intelligence in Education

AMI

Augmented Multy-Party Interaction

ASR

Automatic speech recognition

BART

Bidirectional and Auto-Regressive Transformers

BERT

Bidirectional Encoder Representations from Transformers

CRF

Conditional random field

DA

Domain adaptation

GPT

Generative Pre-trained Transformer

GRU

Gated recurrent unit

ICSI

International Computer Science Institute

IDF

Inverse document frequency

LSA

Latent semantic analysis

LSTM

Long short-term memory

MLM

Masked language model

NER

Named-entity recognition

NLP

Natural language processing

NSP

Next sentence prediction

POS

Part of speech

RNN

Recurrent neural network

ROUGE

Recall-Oriented Understudy for Gisting Evaluation

SOTA

State of the art

SVD

Singular Value Decomposition

TF

Term frequency

TF-IDF

Term frequency/inverse document frequency

Chapter 1

Introduction

> *"The basic pleasure in the phonetic elements of a language and in the style of their patterns, and then in the higher dimension, pleasure in the association of these word-forms with meanings, is of fundamental importance. This pleasure is quite distinct from the practical knowledge of a language, and not the same as an analytic understanding of its structure. It is simpler, deeper—rooted, and yet more immediate than the enjoyment of literature."*
>
> *J.R.R. Tolkien, "English & Welsh", Oxford, 1955*

Over the past years, thanks to many technological innovations, the teaching and learning process has changed. With the introduction of open online courses and e-learning platforms [1] learning-related data experienced a growth.

Distance learning introduces changes in the way students are educated and modifies the roles and the activities of instructors. According to [2] distance learning requires a disproportionate amount of effort from instructors: about 54% of the time spent by an instructor is devoted to create the course content. In [3] the authors discovered that 38 work roles were perceived as important tasks in the distance educators' job, and in the top-10 positions of this list it is possible to find the designer of subject material, the writer of subject material and the facilitator of the learning of subject material. The authors of [4], after reviewing the major studies on stress triggered by the increase of tasks, found that there was minimal research evidence in managing stress among academic staff in higher education institutions.

In parallel, there has been significant growth in the scientific literature concerning the application of AI in education Artificial Intelligence in Education (AIEd) [5, 6, 7]. Along with the rapid development of machine learning and deep learning approaches, novel techniques seek to analyze the vast amount of data obtained from the teaching and learning process.

AIEd supports learning activities in both traditional classes and workplaces by combining AI along with, for example, education, psychology, linguistics, and neuroscience, aiming at stimulating and advancing the development of AI-driven educational applications featuring flexibility, content personalization, and effectiveness [8]. According to [8], the number of AIEd publications showed an exponential trend over the studied years: more than 74% of currently available literature was published between 2012 and 2019.

Among those data, there are the video lectures: the analysis of their content and the extraction of the relevant portions can be extremely useful for several applications according to [9, 10]. For example, a more compact representation of video lectures allows to improve and facilitate the research of a specific content both for students and for algorithms, improving content accessibility.

An immediate and practical application can be the following: it is common to see, for free on-line courses the video content accompanied by a brief description of the lecture. This can help students to conduct faster and more accurate researches based on their necessity. With a brief description, a student can immediately understand the topic of the lecture and therefore decide if it is of his concern.

In alternative, by extracting the most relevant parts of a video lecture it is possible to speed up algorithmic researches. Assuming to have a browser capable of extracting a specific video lecture given a query of a student, the algorithm could process a great amount of video/text data and retrieve the most pertinent results. This operation should be completed in few milliseconds.

Therefore, to accomplish both possibilities, this study analyzes the feasibility of applying an abstractive summarization algorithm to create brief summaries of video lecture transcripts.

Authors in [11] review some implementation of automatic text summarization in the learning context from the year 2010 to 2020, mostly based on extractive summarization [12] (focused on extracting insights from forum discussions [13, 14, 15] or from educational material and video lectures [10, 16, 17]), but none of them proposes abstractive summarization

These techniques derive from Natural Language Processing (NLP) field and their goal is to generate a new text shorter than the one in input. They are different from the extractive summarization one, where the summary is generated starting from the most relevant sentences. In the video lecture context abstractive methods are preferable over extractive summarization methods, as in the latter case the text generated would be composed by a series of sentences not connected to each other and unpleasant from a readability standpoint.

The starting point of this work are the output texts generated by Automatic Speech Recognition (ASR) system that converts audio into text. As will be extensively described in the next chapters, this step may introduce some challenges, as the ASR systems are typically affected by errors in transcription and generate

an unpunctuated text.

For this reason, the first step of the analysis approaches the punctuation restoration problem. For this task different state-of-the-art deep learning models such as Bidirectional Recurrent Neural Networks and Transformer architectures are compared and evaluated. Then, in second instance, the dissertation explores different deep learning architectures for abstractive summarization and their adoption in the educational domain.

The most challenging issue that affects this second step is represented by the length of the video lecture transcripts, which is typically longer than plain text length.

Moreover, the lack of available datasets in the educational context, and more in general, the limited amount of data with similar characteristics, require a preliminary assumption: the most similar setup in the literature is represented by the meeting domain and therefore, the adoption of models design for this context would be beneficial also in the video lecture domain. In this field, there are different studies, architectures and datasets available for abstractive meeting summarization.

Besides state-of-the-art meeting summarization models, a novel method that merges abstractive and extractive summarization tasks is proposed: firstly, the unsupervised extractive algorithm reduces input text length according to the abstractive summarization model limit, and this reduced version is fed into the abstractive model, which returns the final summary.

An ablation study is dedicated to evaluate several aspects: the impact of automatic punctuation on the summarization step, the implications given by the different subjects in MIT courses, and most importantly, it measures the robustness of all the models in the educational domain. The analysis evaluate state-of-the-art models robustness with respect to domain shift and compare these results to the ones obtained with the proposed model.

This approach, with respect to existing models in the meetings domain, reveals greater generalization capability and a good behaviour in domain adaptation.

The thesis is organized as follows:

- The first chapter introduces the objectivewo of this thesis work, highlighting the motivation of this research;
- The second chapter revises some affirmed natural language processing models for language modeling and for sequence labeling;
- The third chapter describes all the datasets involved in the pipeline proposed;
- The fourth chapter is dedicated to the description of state-of-the-art models in both in punctuation restoration task and summarization task; it also dives into the transfer learning problem and the domain adaptation fields;

- Chapter five is devoted to the description of the methods proposed for solving this task;
- The sixth chapter describes the experiments conducted for this problem, and discusses about the results obtained;
- The last chapter presents a recap of the study and discusses about some conclusions and future development.

Chapter 2

Fundamentals of Deep Natural Language Processing

The term natural language processing (NLP) is the task of processing large amount of natural language data, for whatever purpose, regardless of the processing depth. The term *natural language* stands for the languages that it is used in daily life, different from the formal language. The term *Computational Linguistics* may appear to be a synonymous, but in reality differs from NLP in the fact that in linguistics computational methods play a supporting role [18]. In contrast, natural language processing focuses on the design of algorithms for processing human languages.

In linguistics the study of the language is crucial, while in natural language processing a successful algorithm is determined on the basis of how well the task is accomplished, even regardless any linguistic insights.

This chapter covers the main themes that involve the usage of NLP techniques and examines two particular applications treated by the objective of this thesis:

- The sequence labelling problems: the punctuation restoration task is a category of sequence labelling task, consequently, the most relevant architectures are presented;
- The language modelling problems: the ability of modelling a language is a requirement for producing the summarization of a text, making a distinction among contextualized and contextual-independent language models.

2.1 Language models

Given a series of word tokens $w_1, w_2, ..., w_M$ with $w_m \in \mathcal{V}$, where $\mathcal{V}$ represents a discrete vocabulary, a language model assigns a probability distribution over a series of words named $p(w_1, w_2, ..., w_M)$.

These models have shown to be particularly effective in tasks such as machine translation, text summarization or dialogue generation, as they are able to produce word sequences. The main challenge is given by the fluency of the output sequences generated.

2.1.1 Classic models

Word2Vec

In 2013, Mokolov et Al. [19] proposed a novel method for representing words of a text. By the means of large datasets, this architecture is able to compute continuous vector space representations of words by learning the semantic relationships amongst them, based on their context.

In the paper, the authors proposed two architectures: the *Continuous Bag-of-Word* model and *Continuous Skip-gram* model.

In the Continuous Bag-of-Word architecture, the model predicts a center word of a window of surrounding context words. As stated by the authors, is similar to a Feedforward Neural Net Language Model [20], and consists in an input layer, projection layer and output layer. In the input layer the N words are encoded with a 1-of-V (V is the vocabulary size) coding, one-hot encoding vectors of the context words. The input layer is followed by the projection layer, shared for all the words, in which a function g maps the input vectors to a hidden vector. Then, this vector is averaged element-wise and forwarded to the output layer, which returns the probability distribution over the vocabulary.

The Continuous Skip-gram model, instead of predicting the current word based on the context, tries to predict words within a certain range before and after the center word. In this case the current word is the input of the model, turned into a one-hot representation, that flows into the hidden layer and an output layer, which produces the probability vector of each context word to predict. The objective of the Skip-gram model is to maximize the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \tag{2.1}$$

where $w_1, w_2, ...w_T$ are the sequence of training words of cardinality T, c is the number of words before and after the central word.

The two models have two drawbacks: only the weights corresponding to the target word might get a significant update, and the calculation of the final probabilities using the softmax in the output layer is quite an expensive operation.

To mitigate these problems, the authors in [21] propose some modifications. Among these, they introduce negative sampling: instead of trying to predict the probability of being a context word for all the words in the vocabulary, the model tries to predict the probability that some training sample words are part of the context or not. The objective function can be written as follows:

$$\log \sigma(v'^{T}_{w_O} v_{w_I}) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)}[\log \sigma(-v'^{T}_{w_i} v_{w_I})] \tag{2.2}$$

The first term tries to maximize the probability of occurrence for actual words w_O that lie in the context window, w_I. The second term tries to iterate over some random words w_i that do not lie in the window w_I (so they are drawn from a noise distribution $P_n(w)$) and minimize their probability of co-occurrence.

The authors also stated that the distribution of words in a corpus may not be uniform and some words may be rarer than others. So, to counter the imbalance between the rare and frequent words, they propose the subsampling approach. It consists of discarding words with probability:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{2.3}$$

where t is a threshold and $f(w_i)$ is the frequency of word w_i. A schema of the two models is depicted in Figure 2.1.

GloVe

In 2014 researchers from Standford University introduced GloVe [22]. The name stands for Global Vectors and derives from the fact that the global corpus statistics are captured directly by the model.

The training objective of GloVe is to learn word vector representations such that their dot product is close to the logarithm of the words' probability of co-occurrence. Mathematically, the model seeks to minimize the expression:

$$J = \sum_{i,j=1}^{V} f(X_{i,j})(w^T \tilde{w} + b + \tilde{b} - \log X_{i,j})^2 \tag{2.4}$$

where V is the size of the vocabulary, $f(X_{i,j})$ is a weighting function to choose, $X_{i,j}$ is the number of times word j occurs in the context of word i and where w are word vectors and $\tilde{w}$ are the context vectors. The function f is decided a priori and should respect the following properties:

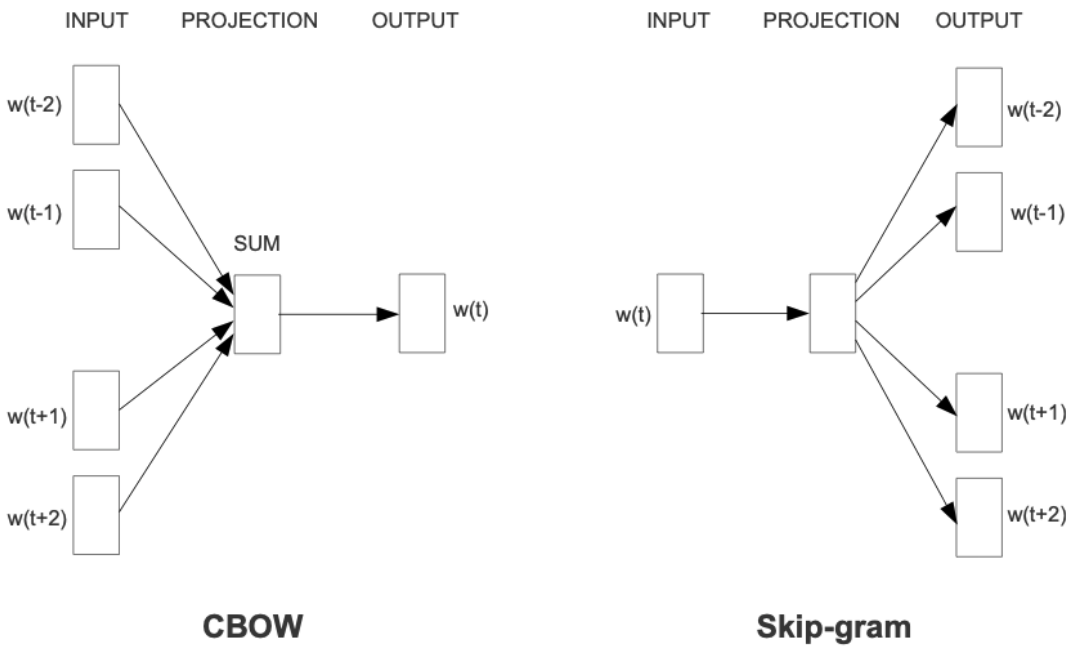

Figure 2.1: Continuous Bag-of-Word and Continuous Skip-gram model architectures presented in [19].

- $f(0) = 0$;
- $f(x)$ should be non-decreasing;
- $f(x)$ should be relatively small for large values of x.

such as:

$$f(x) : \begin{cases} (x/x_{max})^{\alpha} & \text{if} \quad x < x_{max}, \\ 1 & \text{otherwise} \end{cases}$$

In short, the model is a log-bilinear regression model for the unsupervised learning of word representations. It outperformed other models on word analogy, word similarity, and named entity recognition tasks.

2.1.2 Contextualized language models

N-gram language models

To compute the probability of a sequence of tokens a simple approach may be to compute the relative frequency estimate given by the ratio of the count of occurrence of each sequence divided by the total number of possible sentences ever spoken. But this approach, despite its ability to produce an unbiased estimator, can generate high variance due to the large number of possible n-gram that we have in a language (even with a relatively small vocabulary) and this causes relative frequency estimates close to zero. N-gram language models come to tackle this issue by computing the probability of a sequence as:

$$\begin{aligned}\mathbb{P}[\boldsymbol{w}] &= \mathbb{P}[w_1, w_2, ..., w_M] \\ &= \mathbb{P}[w_1] \times \mathbb{P}[w_2|w_1] \times \mathbb{P}[w_3|w_2, w_1] \times ... \times \mathbb{P}[w_M|w_{M-1}, ..., w_1] \end{aligned} \tag{2.5}$$

In particular, the approximation:

$$\mathbb{P}[w_m|w_{m-1}, ..., w_1] \approx \mathbb{P}[w_m|w_{m-1}, ..., w_{m-n+1}] \tag{2.6}$$

This simplification is given by the fact that the word w_m is conditionally dependent only on the previous n words, making the joint probability equal to:

$$\mathbb{P}[w_1, ..., w_M] \approx \prod_{m=1}^{M} \mathbb{P}[w_m|w_{m-1}, ..., w_{m-n+1}] \tag{2.7}$$

The cardinality of the event space is then equal to V^n, and so depends on the order of the n-gram and the decision of n affects the bias-variance complexity trade-off.

Smoothing, backoff and interpolation

Large values of n allow taking into account longer dependencies with the consequence of increasing the variance. To counteract this issue a solution is *smoothing*, that consists of adding some counts:

$$\mathbb{P}_{smooth}(w_m|w_{m-1}) = \frac{count(w_{m-1}, w_m) + \alpha}{\sum_{w' \in \mathcal{V}} count(w_{m-1}, w') + V\alpha} \tag{2.8}$$

The effective counts, given by $c_i^* = (c_i + \alpha)\frac{M}{M+V\alpha}$ where c_i is the count of the event i, $M = \sum_{i=1}^{V} c_i$ is the total number of tokens in the dataset and the discount for each n-gram is given by:

$$d_i = \frac{c_i^*}{c_i} = \frac{(c_i + \alpha)}{c_i}\frac{M}{(M + V\alpha)} \tag{2.9}$$

An alternative can be the *backoff*: instead of assigning the same value for each unseen n-gram, it is possible to consider a lower-order language model. A similar idea is realized in the *interpolation* approach, in which the probability of a word in a context is given by the weighted sum of its probability across shorter contexts, i.e:

$$\begin{aligned}\mathbb{P}_{interpolation}(w_m|w_m, w_{m-1}, w_{m-2}) = \\ = \lambda_3\mathbb{P}_3^*(w_m|w_{m-1}, w_{m-2}) + \lambda_2\mathbb{P}_2^*(w_m|w_{m-1}) + \lambda_1\mathbb{P}_1^*(w_m)\end{aligned} \quad (2.10)$$

with $\sum_{n=1}^{n_{\max}} \lambda_n = 1$.

Recurrent neural network language models

N-gram language models have been supplanted by Recurrent Neural Networks (RNN) [23] because these models are able to capture longer dependencies and at the same time remain computationally tractable. In this case, the task of word prediction becomes discriminative and the main intent is to determine the probability of a word given its context, that depends on the previous words:

$$\mathbb{P}(w|\boldsymbol{u}) = \frac{\boldsymbol{\beta_w} \cdot \boldsymbol{v_u}}{\sum_{w' \in \mathcal{V}} \exp(\boldsymbol{\beta_{w'}} \cdot \boldsymbol{v_u})} \quad (2.11)$$

where $\boldsymbol{\beta_w} \cdot \boldsymbol{v_u}$ represents a dot product between two K dimensional vectors. The natural language model can be built from a recurrent neural network by updating the context vectors while moving through the sequence. Given $x_m \widehat{=} \phi_{w_m}$ as the word embedding of the words w_m, h_m, the contextual information at position m is:

$$h_m = \text{RNN}(\boldsymbol{x_m}, \boldsymbol{h_{m-1}}) \quad (2.12)$$

Then the RNN language model can be defined as:

$$p(w_{m+1}|w_1, w_2, ..., w_m) = \frac{\exp(\boldsymbol{\beta_{w_{m+1}}} \cdot \boldsymbol{h_m})}{\sum_{w' \in \mathcal{V}} \exp(\boldsymbol{\beta_{w'}} \cdot \boldsymbol{h_m})} \quad (2.13)$$

Thanks to the recurrent operation, we can consider all the information about the sequence when processing a word at time m, without limiting the computation to a certain context length. The parameters of the models representing the parameters of the conditional distribution are updated via *backpropagation through time*. The gradients at time m depend on all the previous gradients $n < m$ and the size of the computational graph depends on the input length.

Gates in recurrent neural networks

Repeated application of non-linear functions may cause the exploding gradients or the vanishing gradients problem. The former can be addressed by clipping the gradients over a certain threshold; the latter, being a more complicated issue, requires changes in the architecture. In particular, two variants of RNN have been developed:

- Long short-term memory (LSTM);
- Gated recurrent unit (GRU);

The LSTM are equipped with a *memory cell* c_m which contributes to form the next states h_m. The advantage is that the memory cell does not pass through a squashing function and the information can flow through the network without vanishing.

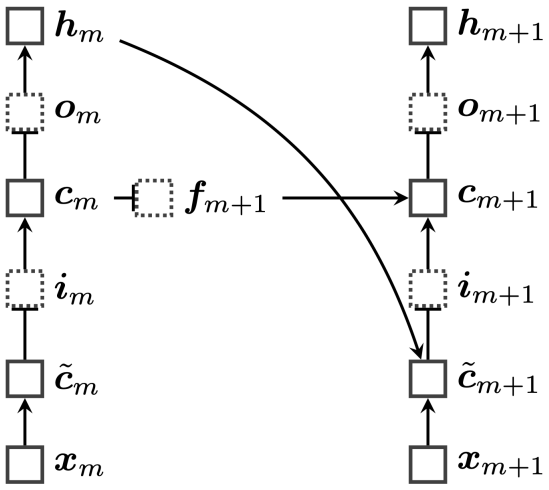

Figure 2.2: Long short-term memory structure taken from [18], gates are displayed in dotted edge boxes.

The gates are functions that control how much information propagates through the network: they compute the Sigmoid of the linear combination of inputs and the previous hidden states. There are different kinds of gates:

- Forget gate: $\boldsymbol{f}_{m+1} = \sigma(\boldsymbol{\Theta}^{(h\to f)}\boldsymbol{h_m} + \boldsymbol{\Theta}^{(x\to f)}\boldsymbol{x}_{m+1} + \boldsymbol{b}_f)$
- Input gate: $\boldsymbol{i}_{m+1} = \sigma(\boldsymbol{\Theta}^{(h\to i)}\boldsymbol{h_m} + \boldsymbol{\Theta}^{(x\to i)}\boldsymbol{x}_{m+1} + \boldsymbol{b}_i)$
- Output gate: $\boldsymbol{o}_{m+1} = \sigma(\boldsymbol{\Theta}^{(h\to o)}\boldsymbol{h_m} + \boldsymbol{\Theta}^{(x\to o)}\boldsymbol{x}_{m+1} + \boldsymbol{b}_o)$

The cell memory is given by:

$$\boldsymbol{c}_{m+1} = \boldsymbol{f}_{m+1} \odot \boldsymbol{c}_m + \boldsymbol{i}_{m+1} \odot \tilde{\boldsymbol{c}}_{m+1} \tag{2.14}$$

where $\tilde{c}_{m+1} = \tanh(\boldsymbol{\Theta}^{(\boldsymbol{h}\to\boldsymbol{c})} h_m + \boldsymbol{\Theta}^{(\boldsymbol{x}\to\boldsymbol{c})} \boldsymbol{x}_{m+1})$ is the updated candidate. The output of each state is given by $\boldsymbol{h}_{m+1} = \boldsymbol{o}_{m+1} \odot \tanh(\boldsymbol{c}_{m+1})$. The memory cell of the previous state contributes to the memory of the subsequent state, and, indirectly, also to the subsequent output, hence avoiding the gradient vanishing. The complete structure is displayed in Figure 2.3

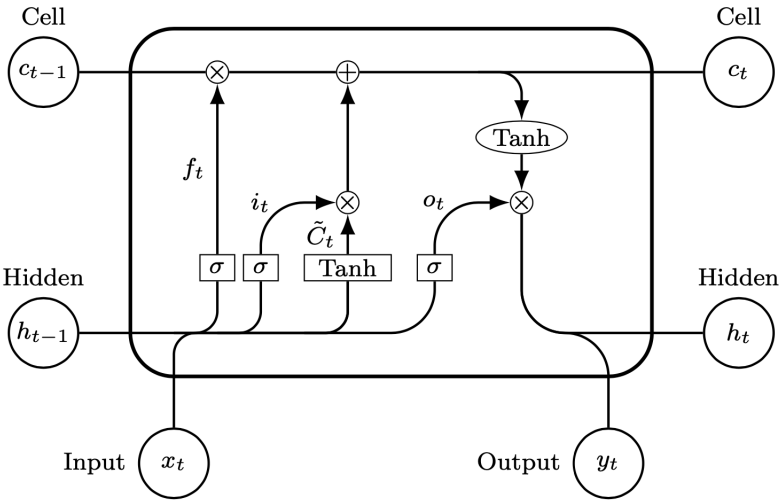

Figure 2.3: LSTM synthetic structure.

Transformer-based language models

Transformers architectures [24] and their variants have been widely explored in the machine translation task as they have been proven to have better performance with respect to recurrent models and more parallelizable, requiring significantly less time for training.

The input sequence $\boldsymbol{x}_i$ is converted into an embedding sequence $\boldsymbol{e}_i$ and a positional encoding is added in order to quantify the position of each token in the sentence [1]. For each position *pos* the positional encoding of dimension i can be computed as:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{1000^{2i/d_{\text{model}}}}\right) \tag{2.15}$$

[1]The transformer processes the complete sequence of tokens in parallel, eliminating any form of information regarding the position of each word in the sentence

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{1000^{2i/d_{\text{model}}}}\right) \tag{2.16}$$

The embeddings are then encoded into the *multi-head self-attention layer* of the encoder. This layer takes in input Q, K and V matrices, respectively the query, the key and the value matrices, and for each head computes the attention as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.17}$$

where W_i^Q, W_i^K, W_i^V are parameters to learn, and the attention operation is given by:

$$\text{Attention}(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \tag{2.18}$$

The multi-head attention concatenates the output of each head and passes it to the *feed-forward layer* that consist of two fully connected layers with a ReLU activation function. Between each layer a *residual connection*, connecting the previous input and the current output, helps the network avoiding the vanishing gradient problem (for further details see *Residual connections* Section in Appendix B). This way the self-attention mechanism can be repeated many times.

Different language models have been developed so far. For language modelling/language understanding tasks these models can be trained in three different ways according to [25]:

- In an *autoregressive* fashion: autoregressive models are pretrained on the classic language modeling task, they try to guess the next token having read all the previous ones; these models rely on the decoder part of the transformer architecture previously described; these architectures employ masked attention mechanisms, so that at each position, the model can only look at the previous tokens in the attention heads. At inference time, the decoder output constitutes its next input and the sequence is generated until the stop tag is predicted or up to a certain maximum length. An example is GPT [26].

- In an *autoencoding* fashion: these models rely on the encoder part of the original transformer and use no mask so the model can look at all the tokens in the attention heads. For pretraining, inputs are a corrupted version of the sentence, usually obtained by masking tokens, and targets at random, while the target is the original (not corrupted) sequence. A typical example of a structure that adopts this training is BERT [27].

- *Sequence-to-sequence* training: these models keep both the encoder and the decoder of the original transformer. The encoder is trained with *autoencoding* while the decoder in an *autoregressive* manner. BART model (that will be used for the summarization task) [28] uses this approach for training.

A schema of the typical transformer model is shown in Figure 2.4.

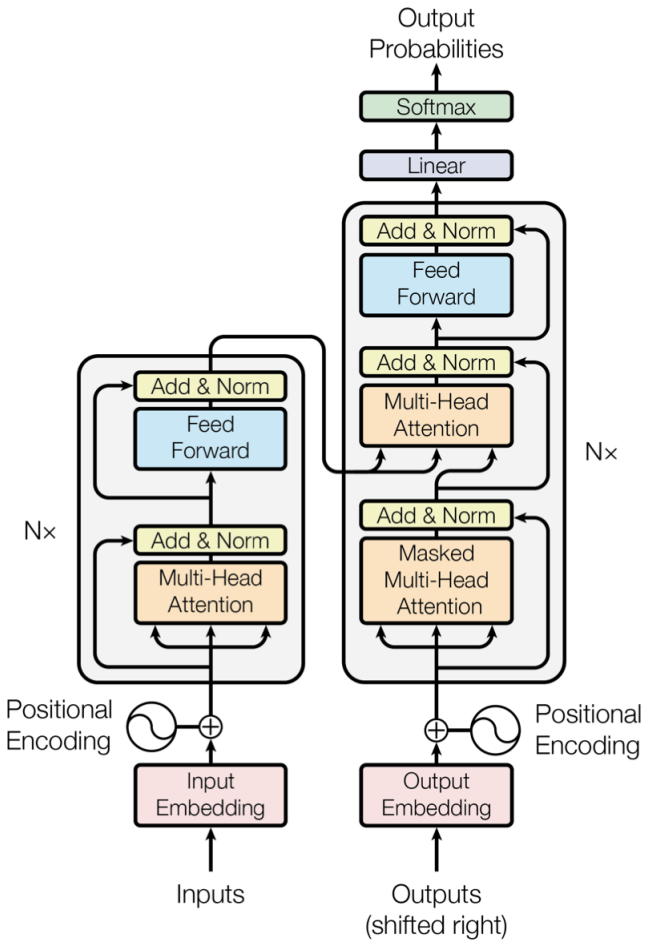

Figure 2.4: Structure of the transformer model presented in [24] for punctuation restoration.

BERT model

BERT model [27], acronym of *Bidirectional Encoder Representation from Transformers*, is a multi-layer bidirectional encoder of the Transformer architecture trained in two steps. Firstly, in a *pre-training* phase, the model is trained on unlabeled data, and later, starting from the pre-trained parameters, the model is *fine-tuned* on a specific task with the labeled data.

More specifically, in the *pre-trained* phase, the authors present two unsupervised tasks for BERT pre-training. The first is the so-called procedure *Masked Language Model* (MLM): a percentage of the input tokens at random is masked and the

model is trained to learn these masked tokens. BERT uniformly selects 15% of the input tokens for possible replacement. Of the selected tokens, 80% are replaced with [MASK], 10% are left unchanged and 10% are replaced by a randomly selected token in the vocabulary.

The second, called *Next Sentence Prediction* (NSP) is represented by a binary classification problem, in which the model is trained to recognize which sentence pairs should be considered contiguous. This second task is particularly suitable in case the main objective is to create an architecture capable of capturing dependencies among sentences.

In the fine-tuning phase the model is trained on a specific downstream task, starting from the set of weights found with the previous step. This task, according to the authors, requires less time for training, as requires few parameters to train for adapting the model to the specific task. BERT is designed to learn bidirectional representations via jointly conditioning on both left and right contexts and consequently more powerful than unidirectional models.

GPT model

In [26] the authors propose an architecture, called GPT (*Generative Pre-Training*), which unlike BERT, is an autoregressive model and outputs one token at a time. As in BERT model, the GPT architecture is trained in two different phases. Again in this architecture, the goal of the unsupervised pre-training is to find a good initialization point for many NLP tasks. It has been discovered that this procedure helps in model regularization, allowing more generalization.

In particular, the authors propose the following objective function:

$$Loss(\Theta) = \sum_i \log \mathbb{P}[w_i | w_{i-1}, w_{i-2}, ..., w_{i-c} | \Theta] \tag{2.19}$$

where Θ are the model parameters, c is the context window. The paper proposes the usage of Transformer architecture, in particular, the decoder part in order to produce the output distribution. The formula (2.19) points out the autoregressive nature of the model, meaning that the decoder is capable of generating new predictions according to the previous outputs. The supervised fine-tuning step is done after the pre-training step, this phase seeks to train again the model in order to assign to each input token a label y_i. The inputs of the specific downstream tasks are arranged into ordered sequences in order to accomplish this second task.

The novelty introduced by GPT-2 [29] is that the authors demonstrate the ability of language models to perform a wide range of tasks in a zero-shot fashion, using a larger dataset in the pre-training phase and adding more parameters.

In particular, the paper proposes what is called *task conditioning*, through which the learning objective should be modified to $\mathbb{P}[\text{output}|\text{input}, \text{task}]$, and the model

is expected to produce different outputs for the same input, for different tasks. The capability of understanding the task allows to avoid the input transformation step latter described (a property called *zero-shot task transfer*).

In GPT-3 [30] the number of parameters increases to 175 B and the model starts recognizing patterns in data that help the model during zero-shot task transfer, becoming *meta-learners*. The GPT-3 model is effective in few-shot context, the setting where the model is fed with a few examples of the task at inference time and the task description but no weight updates are allowed, and one-shot learning (only one example is allowed) or the most extreme zero-shot setting, in which the model only knows the description of the task in natural language.

2.2 Sequence labelling

The task of sequence labelling aims to find the correct discrete label for each word in a sequence. More formally, given a sequence of words $\boldsymbol{w} = (w_1, w_2, ..., w_M)$, a set of possible tags $\mathcal{Y}(m) = \mathcal{Y}^M$ of length M the sequence labelling problem can be formulated as follows:

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{m})} \Psi(\boldsymbol{w}, \boldsymbol{y}) \tag{2.20}$$

where Ψ is a scoring function, $\boldsymbol{y} = (y_1, y_2, ..., y_M)$ is a sequence of predicted tag. The label space grows exponentially with the length M and can be intractable but by restricting the scoring function enables a more efficient inference.

2.2.1 Discriminative models for sequence labelling

The scoring function can be represented as a weighted sum of local features, so:

$$\Psi(\boldsymbol{w}, \boldsymbol{y}) = \boldsymbol{\theta} \cdot f^{(global)}(\boldsymbol{w}, \boldsymbol{y}_{1:M}) \tag{2.21}$$

where $f^{(global)} = \sum_{m=1}^{M+1} f(\boldsymbol{w}_{1:M}, y_m, y_{m-1}, m)$. The estimate of the parameters $\boldsymbol{\theta}$ can be carried out throughout discriminative models such as:

1. *Perceptron*: a perceptron [31] that uses the Viterbi algorithm for an efficient search.

2. *Support vector machine*: an SVM [32] counterpart;

3. *Conditional random fields*: derives from the logistic regression classifier, the probability model is given by:

$$\mathbb{P}[\boldsymbol{y}|\boldsymbol{w}] = \frac{\exp(\Psi(\boldsymbol{w}, \boldsymbol{y}))}{\sum_{\boldsymbol{y}' \in \mathcal{Y}(\boldsymbol{w})} \exp(\Psi(\boldsymbol{w}, \boldsymbol{y}'))} \tag{2.22}$$

For training, the weights $\boldsymbol{\theta}$ are learned via:

$$\arg\min_{\boldsymbol{\theta}} \left\{ \frac{\lambda}{2} ||\boldsymbol{\theta}||^2 - \sum_{i=1}^{N} \boldsymbol{\theta} \cdot f(\boldsymbol{w}^{(i)}, \boldsymbol{y}^{(i)}) + \log \sum_{\boldsymbol{y}' \in \mathcal{Y}(\boldsymbol{w}^{(i)})} \exp\left(\boldsymbol{\theta} \cdot f(\boldsymbol{w}^{(i)}, \boldsymbol{y}')\right) \right\} \tag{2.23}$$

where λ is an hyperparameter.

2.2.2 Recurrent neural network for sequence labelling

Given a recurrent model:

$$h_m = \text{RNN}(\boldsymbol{x_m}, \boldsymbol{h_{m-1}}) \tag{2.24}$$

for $m = 1, 2, ...M$, where $\boldsymbol{x_m}$ is the embedding of token w_m and $\boldsymbol{h_{m-1}}$ is the previous hidden state. A possible application of RNN models in sequence labelling can be constructed as a linear function of hidden states:

$$\psi_m(y) = \boldsymbol{\beta_y} \cdot \boldsymbol{h_m} \tag{2.25}$$

$$\hat{y} = \arg\max_y \psi_m(y) \tag{2.26}$$

that can be converted in probability, obtaining an estimate of $\mathbb{P}[y|\boldsymbol{w_{1:M}}]$, via softmax operation. It is important to point out that simple recurrent models only take into account information coming from precedent tokens, while a Bidirectional Recurrent Neural Network does not ignore subsequent information.

More formally, denoted $\overleftarrow{\boldsymbol{h}}_{\boldsymbol{m}}$ the right-to-left and $\overrightarrow{\boldsymbol{h}}_{\boldsymbol{m}}$ the left-to-right hidden state vectors, these vectors are concatenated forming $\boldsymbol{h_m} = [\overleftarrow{\boldsymbol{h}}_{\boldsymbol{m}}, \overrightarrow{\boldsymbol{h}}_{\boldsymbol{m}}]$, a vector that summarizes the useful information for a specific token w_m coming from the surrounding context. For this reason, after applying a linear function as in the unidirectional case, it is possible to obtain more accurate estimates of the tag sequence.

2.2.3 Transformer-based for sequence labelling

Given the transformer language model described in the previous section (see Section Transformer-based language models for mathematical details of Transformer components), it is possible to adapt these models for the sequence labelling task.

In general, the output of a transformer encoder is the set of features extracted by a pre-trained language models.

Typically, for these tasks *autoencoding* models are employed. These models are trained on language understanding task, so are effectively capable of encoding contextual information and long-range dependencies . Then, the sequence labelling task is performed by adding a classification layer on top of them. This layer takes the concatenation of the last hidden states from the underlying feature extractor.

By modelling these features it is possible to associate at each input sequence $\boldsymbol{x_{1:M}}$ a series of label $\hat{\boldsymbol{y}}$, one for each token, obtained through the estimate of $\mathbb{P}[\hat{\boldsymbol{y}}|\boldsymbol{x_{1:M}}]$. This approach reveals to be particularly effective in numerous sequence labelling tasks such as POS/NER recognition, as these models are capable of leveraging pre-trained language models and adapt them for a specific task.

The training of these models employs the negative log-likelihood loss, described as follows:

$$\text{Loss}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_{i=0}^{N} y_i \cdot \log(\mathbb{P}[\hat{y}_i|x_i]) \tag{2.27}$$

In this thesis, a particular architecture of transformer is studied for the punctuation restoration task, and compared with a recurrent architecture.

Chapter 3

Datasets

The scope of this thesis is to produce a synthetic representation of Politecnico video lecture transcripts. The data provided by the university consist of raw transcriptions of video lecture, and requires some pre-processing steps. Furthermore, in order to evaluate the goodness of the chosen approaches, some datasets, well know in literature are employed. These datasets evaluate the impact of the techniques applied both in preprocessing and in summarization task. More in detail, the Politecnico video lecture transcriptions miss punctuation marks, and the restoration is conducted with models trained on IWSLT [33] dataset. The summarization task instead involves CNN [34] and SAMSum [35] dataset on which models are pre-trained, and AMI [36] and ICSI [37] datasets used for model finetuning.

Furthermore, for a fair evaluation of the summarization task, based on real video lecture, the dataset EduSum is employed.

3.1 The IWSLT dataset for punctuation restoration

The lack of punctuation marks in video lecture transcripts requires a pre-processing step which allow to obrain a text punctuated.

A well-known dataset for punctuation restoration is given by the IWSLT dataset [33]. This dataset comes from the International Workshop on Spoken Language Translation (IWSLT), an annual scientific workshop evaluation focused on the automatic translation of public talks [1].

It is composed of four typologies of track:

- Automatic speech recognition (ASR) tracks, the automatic transcription of talks from audio in English of the TED talks to text;
- Spoken language translation (SLT) tracks for the translation of the automatic English transcriptions of the talks (ASR output) to French;
- Machine translation (MT) that consists in the translation of the manual transcriptions and translations of the talks from English to French, from Arabic to English, and from Chinese to English;
- System combination (SC) track for combining the English ASR outputs and/or the MT outputs in English and French.

For the punctuation restoration task, the training and validation datasets used are derived from the conference of 2012 (IWSLT2012) and consist of 2.1M and 296k words, respectively. The test set is taken from the test data of IWSLT2011 ASR dataset. The ground truth presents four labels including three punctuation marks: "COMMA" (that includes commas, colons and dashes), "PERIOD" (including full stops, exclamation marks and semicolons), "QUESTION" (for question marks only), and "O" for any other tokens.

[1] Website at: http://hltc.cs.ust.hk

3.2 The AMI and ICSI dataset for meeting summarization

The AMI (Augmented Multy-Party Interaction) dataset is the outcome of 100 hours of meeting recordings.

The data collected include audio tracks, video recordings, slides projected during the presentations, the input in the electronic whiteboard. The data has been collected with different instruments and placements in the room, so the audio tracks produced vary

Moreover, datasets are annotated with speech transcription, named entities, dialogue acts, topics involved, abstractive and extractive summaries, emotions, head and hand gestures, location of individuals and postures, and the focus of attention and can be adopted in tasks of various nature.

These meetings are attended by a series of speakers, each defined by a different role. Among those there are:

- The project manager (PM);
- The marketing expert (ME);
- The user interface designers (UI);
- The industrial designer (ID).

The ICSI (International Computer Science Institute) dataset [37] contains 70 hours of meeting recordings that occurred in the International Computer Science Institute and has a similar structure to the AMI dataset. Unfortunately, extractive summaries were no more available. A summary table of the dataset is reported in Table 3.1.

Dataset	Number of meetings			Input length		Summary length	
	Train	Valid	Test	Words	Token	Words	Token
AMI	100	17	20	3156	4081	280	321
ICSI	43	10	6	6228	7913	466	576

Table 3.1: Summary of statistics of AMI and ICSI dataset.

3.3 The EduSum video lectures dataset

The proposed dataset seeks to address the lack of a specific dataset for abstractive summarization of video lecture transcripts. Meeting dataset differs from the video lecture dataset for several reasons:

1. Typically each video lecture is held by a single person, a professor, researcher or a Ph.D. student, that carries out the lesson in an autonomous way. There might be interruptions or student questions, but in general, they are not included in the recording because of technical reasons;

2. Depending on how the transcriptions have been taken the error rate may vary. According to [38] the word error rate of the ASR system transcriptions is respectively 36% and 37% for AMI and ICSI dataset.

Given these differences, and without any ground truth available for the Politecnico video lectures, a novel dataset, called *EduSum*, is built. This dataset is taken from the MIT OpenCourseWare website [2] that publishes all MIT course content. The website is completely open and available: it contains the video lectures, the transcriptions, the supporting material with slides and some students notes. The courses have different nature: from more scientific fields such as Chemistry to humanistic subjects, such as Cinema.

Among all the courses, the courses are selected according to the following criteria. Firstly, only the complete transcripts, correctly annotated even with the punctuation marks are selected: the reason is that, in the following discussions, models for punctuation restoration task are evaluated according to the results obtained with this dataset as well, to better estimate errors in Politecnico video lecture data. Secondly, only lectures that present the *Description* section are considered: it represents the abstractive summary of the video lesson.

The courses chosen for this work are:

- STS-081 [39], Innovation Systems for Science, Technology, Energy, Manufacturing, and Health: a course that that study the science and technology innovation system with a particular attention in fields like energy, computing, advanced manufacturing, and health sectors;

- 21L-011 [40], The Film Experience: the course is focused on the anthropological and historical artifacts that characterize classic films.

- 5-111SC [41], Principles of Chemical Science: is an introductory course to the chemistry of biological, inorganic, and organic molecules.

[2]Link at: https://ocw.mit.edu/about/

- 6-006 [42], Introduction to algorithm: the course covers algorithmic paradigms, and data structures used to solve these problems.
- 6-S897 [43], Machine Learning for Healthcare: this course presents some applications of machine learning approaches in the healthcare sector.
- 7-91J [44], Foundations of Computational and Systems Biology: the course merges biology, engineering, and computer science.
- 15-S12 [45], Blockchain and Money: the course covers the Bitcoin and an understanding of the economic and technical fundamentals of blockchain technology.

The mean length of the transcripts is 10170 +/- 3688 tokens and the mean summary length is about 100 +/- 60.

An extract of a video lecture of this dataset is the following:

> *All right. Let's dive right into education. And, you know, this is the other side of the innovation equation, right? We've talked in terms of institutions and linking and connecting institutions, and we've talked a lot about R&D and the R&D system, but you know, from the first class on, Romer taught us that the talent base is a very critical consideration in innovation. And how do you build up the talent base? That's essentially our set of tests today. So, first, Norm Augustine. I wanted to acquaint you with Norm Augustine, because in the science and technology community, he's known as St. Augustine. And he is just an incredible stand-up figure on issues like the importance of federal R&D investment. He was chairman for a lengthy period of time of Lockheed Martin. He won the President's Medal of Technology. Just a noted innovator, himself, but also a true expert on R&D issues, R&D policies. Unfailingly helpful and willing to volunteer for whatever task the National Academies or in many cases MIT, or other organizations kind of need help and advice from a real senior statesman, Augustine has always been willing to step up to the plate. So he's kind of a remarkable figure. He led the rising against the Gathering Storm report back in 2000– around the early 2000s timetable– along with people like Chuck Vest, and a really noted community of other experts, that made the argument for a very significant R&D increase for the physical science agencies. Around that time, we had been doubling NIH. This report made the case for doubling the physical science-based R&D areas, as well. And it was a very influential and important report. He was one of the real leaders of it, and he played an important role, along with Chuck Vest, in helping persuade the latest Bush administration– hi, Karen– to adopt its recommendations. So there was a period of time*

of ongoing R&D increases for the physical science agencies. So the report had a result [...]

and the corresponding summary:

Class 11 will consider overall science education trend data, and the reasons for the decline in college level science graduates, as well as graduate students, particularly in the physical sciences. A recent summary by Norman Augustine for the National Academies summarizes US talent gap problems. The class will then examine studies by economists Paul Romer and Richard Freeman in fixing the basis for these trends. It will also review an economic study suggesting a link between education attainment and growing income differentiation. The class will review an appeal by economist William Bamol for a new kind of "innovation education." The class will also discuss reforms to teaching science education and new approaches to IT-based education models.

3.4 The Politecnico video lectures dataset

In this essay, the dataset under analysis is relative to the Electronic and Fundamentals Application course offered at Politecnico di Torino in 2019.

This dataset is composed of a series of text files, one for each class, that report the ASR transcriptions of the course. The analyzed course is offered in English. The most evident issue of this dataset is the lack of punctuation: the transcriptions do not contain any punctuation marks and in order to employ abstractive summarization technique it is necessary to restore it. This is due to the ASR system used, which, as the majority of ASR systems, merely transcribes the speakers' words. For this reason, in the following chapters, the thesis describes how to perform punctuation restoration task via deep learning techniques.

The transcriptions were generated with automatic services, which, according to [46] and [47], are typically affected by errors. The papers compare the best ASR products available at their time of publication with different datasets, and assert that ASR systems, on average, are affected by a Word Error Rate [3] of about 40%. The sources of errors are mainly represented by:

- technical recording issues;
- preprocessing of the original audio file that may affect audio quality (for example due to downsampling);
- Noisy environment;
- Poor speaker articulation;
- Presence of unusual words in the discourse.

As far as the video lecture dataset is concerned, the last point takes on particular relevance, as the analyzed subject is rich in formulas. Furthermore, most of the discourses are characterized by terms that belong to the Electronic lexicon and lack punctuation marks.

The Electronic and Fundamentals Application dataset is composed of 25 classes, each class contains on average 7585 words with a high variability of 3476 due to the presence of classes much shorter than others. The mean word length is about 4.3 characters; these statistics are in line with ones of other common datasets of ASR transcriptions.

An example is reported below:

[3]The Word Error Rate is represented by the number of substitutions, insertions and deletions over the number of words in the reference text.

so at very low frequency we have the plus 90 due to the zero and then we have in two decades we have the contribution of minus 90 due to the pole and so we reach zero degree after two decades the theoretical behavior starts one decades more before and and one decade after the real behavior starts two decades before and ends two decades after and in the middle we just have 90 minus 45 that means 45 degree also in this case we can easily calculate and implement the amplification factor independently from the cutting frequency of the high pass filter because the gain here depends from minus r2 over r1 while the cutting frequency depends from r1 and the capacitor okay so we have enough degree of freedom to calculate all the components and separate the gain from the cutting frequency the last circuit is very simple and is just the combination of the low pass filter and the high pass filter so as you can see we have now uh z1 a full z1 impedance and said to impedance here we have the series of a capacitor and of a resistor and here we have the parallel of a capacitance and of a resistor and the amplification b u over v e is always minus z 2 over z1 always the same and the full expression became just the combination of the previous two expression [...]

Chapter 4

State of the art

4.1 Punctuation restoration

The main problem of using text summarization algorithms in speech transcript summarization tasks is determined by the lack of punctuation. Restoring the punctuation in automatic speech recognition system outputs greatly improves not only the readability, but also subsequent processing, such as text summarization.

Many attempts have been made to predict punctuation marks automatically. These approaches can be roughly divided into three categories in terms of applied features:

- prosodic features (properties of syllables and larger units of speech, including linguistic functions such as intonation, tone, stress, and rhythm): typically prosody based models have been proven to be more robust to ASR system errors according to [48] ;
- lexical features (single word, a part of a word, or a chain of words that forms the basic elements of a language vocabulary);
- the combination of the previous two features based methods.

The original audio signal of the speech would be very informative for this kind of task: pause between words, pitch, tone and intensity of the voice constitute an important source of information to determine changes in the speech.

In this work only the transcripts will be considered available, and all the techniques that restore punctuation by the means of the original audio signal will not be further described. The punctuation restoration task is a kind of the sequence labelling task presented in *Sequence labelling* Section.

The first attempts employ several methods such as n-gram models [49], Conditional Random Fields [50] and transition-based dependency parsing, deep and convolutional neural networks.

Othes suggest considering the punctuation restoration task as a machine translation task, translating from unpunctuated text to punctuated text, and therefore a kind of language modelling task, presented in Section *Language models*. All the presented methods have been trained and evaluated on the IWSLT dataset.

4.1.1 Previous approaches

The *n*-gram language model

In [49] the authors present a novel method for punctuation restoration tasks based on language models with n-gram. A language model represents a probability distribution over sequences of words. In order to predict the next token, the model relies on two assumptions: the conditional independence of the tokens and the Markov assumption that only the most recent tokens $(n-1)$ are relevant. By estimating the relative likelihood of different phrases of a punctuated text, the language model is then used for inserting the punctuation in a non-punctuated text.

The authors took inspiration from [51] and adapt the model proposed for the punctuation restoration task. So given a source-text without punctuation marks $\boldsymbol{u}$, they want to automatically produce a punctuated text $\hat{\boldsymbol{p}}$ such that:

$$\hat{\boldsymbol{p}} = \arg\max_{\boldsymbol{p}} \sum_{m_1}^{M} \lambda_m h_m(\boldsymbol{p}, \boldsymbol{u}) \tag{4.1}$$

where $h_m(\boldsymbol{p}, \boldsymbol{u})$ is a set of feature functions with cardinality M and λ_m is a set of weights.

Given a $w_1^L = (w_1, ..., w_L)$, a string of L tokens over a fixed vocabulary, the model proposed defines the probability of w_1^L as:

$$\mathbb{P}[w_1^L] = \prod_{i=1}^{L} \mathbb{P}[w_i | w_1^{i-1}] \approx \prod_{i=1}^{L} \mathbb{P}[w_i | w_{i-n+1}^{i-1}] \tag{4.2}$$

Defining a substring w_i^j of w_1^L and its frequency $f(w_i^j)$, i.e. the number of times in which the substring occurs in a *training data*, the maximum likelihood probability estimate of an n-gram is given by:

$$r(w_i | w_{i-n+1}^{i-1}) = \frac{f(w_{i-n+1}^{i})}{f(w_{i-n+1}^{i-1})} \tag{4.3}$$

that represents the relative frequency of the n-gram.

The problem of this approach is that typically the training dataset is sparse, and the denominator of (4.3) can assume negative values. Furthermore, the results can be improved by increasing the number n, which increases the sparsity.

The main argue for this approach are the following: the n-gram language model is only able to capture surrounding contextual information, while for punctuation restoration tasks, modeling of longer-range dependencies is necessary. In the second instance, it is not robust to manage cases in which noisy or out-of-vocabulary words appear frequently.

The Conditional Random Fields

To overcome the highlighted problems of the n-gram language models, the authors in [50] define the Conditional Random Fields (CRF) model as a discriminative model of the conditional distribution of the complete label sequence given the observation.

The CRF adopts an undirected graph to model the conditioning relationship between words and labels.

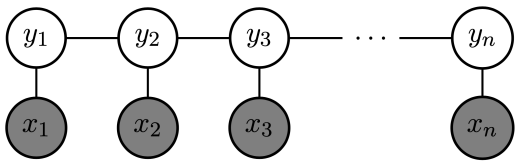

Figure 4.1: Graphical representation of observation $x_1, x_2, ..., x_n$ and labels $y_1, y_2, ..., y_n$ presented in [50]

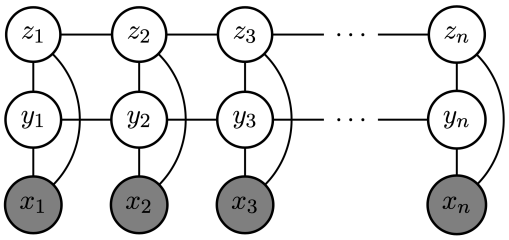

Figure 4.2: Graphical representation of observation $x_1, x_2, ..., x_n$ with two set of labels $y_1, y_2, ..., y_n$ and $z_1, z_2, ..., z_n$ presented in [50]

The punctuation prediction task is the process of assigning a tag to each word: each word can be characterized by an event that defines which punctuation symbol should be inserted (including the possibility of non insertion) after the corresponding word.

The training dataset is composed of a list of words with their corresponding tags that codify the punctuation mark. The features are binary and collect information about the n-gram occurrence surrounding the current word and the position of the current word.

During inference, the most probable sequence of tags is chosen. The problem with this approach is that it only learns a sequence of tags at the individual word level but is not capable of managing sentence level information.

For this reason the authors propose a method that jointly performs sentence segmentation and words level punctuation restoration by assigning two levels of

tags:

- punctuation tag, that acts at word layer, responsible for inserting the punctuation symbol after each word;
- a sentence bound tag, which defines the annotation of sentence boundaries and identifies the sentence type.

The conditional probability of a sequence of label vector $\boldsymbol{y}$ given the observation $\boldsymbol{x}$ is given by:

$$p_\lambda(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \exp \sum_t \sum_{c \in C} \sum_k \lambda_k f_k(\boldsymbol{x}, y_{(c,t)}, t) \tag{4.4}$$

where C represents the set labels sets, $Z(\boldsymbol{x})$ is a normalization factor. The sentence layer tags are able to improve the prediction of the punctuation symbol at word level as it is able to capture long dependencies.

4.1.2 Bidirectional Recurrent Neural Network

In [48] the authors present an architecture named *Punctuator*, that aims to insert punctuation in an unsegmented text by the use of bidirectional recurrent structures in combination with an attention mechanism.

The model is composed of three features:

- The bidirectional recurrent layer: this layer is composed of two unidirectional gated recurrent units that seek to capture the context of each word. The chosen of the GRU structure is motivated by the fact that they are able to capture long-range dependencies being not affected by the vanishing gradient problem, using fewer parameters than a LSTM.
- The attention mechanism [52]: is performed in to determine which parts of the speech are the most relevant to determine the correct position of a punctuation mark.
- The late fusion approach merges the bidirectional recurrent layer output state and the attention mechanism output.

The input sequence of words is encoded in one-hot vector $\boldsymbol{X} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_T)$ that then is processed into an embedding layer followed by the bidirectional layer consisting of two recurrent layers with GRU, one for the forward direction and one for the backward direction.

$$\overrightarrow{\boldsymbol{h}}_t = \text{GRU}(\boldsymbol{x}_t \boldsymbol{W}_e, \overrightarrow{\boldsymbol{h}}_{t-1}) \tag{4.5}$$

The forward bidirectional layer processes the input in the normal order, while the

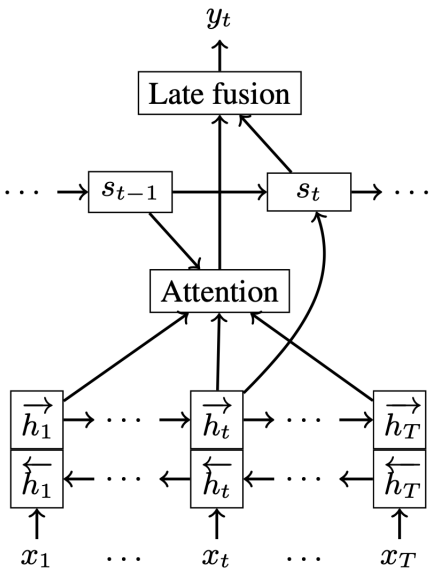

Figure 4.3: Structure of the model presented in [48], called *Punctuator*, a bidirectional RNN with attention mechanism for punctuation restoration.

other bidirectional layer processes the input in the reverse order: the state of a time t will be the result of the concatenation of the hidden states of the two layers.

$$\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}}_t, \overleftarrow{\boldsymbol{h}}_t] \tag{4.6}$$

The bidirectional layer is then followed by a unidirectional GRU layer characterized by an attention mechanism: it processes the bidirectional states sequentially and manages the information of the current position of the word in the text, while the attention mechanism can focus on relevant information on the context. The state of this layer $\boldsymbol{s}_t$:

$$\boldsymbol{s}_t = \mathrm{GRU}(\boldsymbol{h}_t, \boldsymbol{s}_{t-1}) \tag{4.7}$$

and the output of the attention model $\boldsymbol{a}_t$ compose the late fused state $\boldsymbol{f}_t$:

$$\boldsymbol{f}_t = \boldsymbol{a}_t \boldsymbol{W}_{fa} \cdot \sigma(\boldsymbol{a}_t \boldsymbol{W}_{fa} \boldsymbol{W}_{ff} + \boldsymbol{h}_t \boldsymbol{W}_{fh} + \boldsymbol{b}_f) + \boldsymbol{h}_t \tag{4.8}$$

which pass through the output layer that computes the punctuation probabilities $\boldsymbol{y}_t = Softmax(\boldsymbol{f}_t \boldsymbol{W}_y + \boldsymbol{b}_y)$.

The attention mechanism and the late fusion approach

The strength of this model is given by the attention mechanism and the late fusion approach proposed in [52]. In this paper the authors start from a particular structure called RNN Encoder-Decoder architecture. For the Encoder, they consider a Bidirectional RNN and call the forward and backward hidden states concatenation as *annotations* h_j of the word x_j. The Decoder is given by and RNN that computes the hidden state s_i as:

$$\boldsymbol{s}_t = f(\boldsymbol{s}_{t-1}, \boldsymbol{y}_{t-1}, \boldsymbol{c}_t) \tag{4.9}$$

and performs the alignment process for computing the context vector c_i: takes the encoder annotations and computes the energy e_{ij} as:

$$\boldsymbol{e}_{ij} = a(\boldsymbol{s}_{i-1}, \boldsymbol{h}_j) \tag{4.10}$$

where $\boldsymbol{s}_{i-1}$ is the RNN hidden state and $a(\cdot)$ is an alignment model, that scores how well the inputs around position j and the output at position i match. These scores are passed through a softmax function and generate the weights $\boldsymbol{\alpha}_{ij}$ of each annotation $\boldsymbol{h}_{ij}$. The context vectors are the weighted sum of the annotations, where the weights are $\boldsymbol{\alpha}_{ij}$. In the paper of Bahdanau et Al. the authors focus on the machine translation task: the *Punctuator* adapts this structure to increase the model capacity of finding relevant parts of the context for punctuation decisions.

The problem of this approach is given by the fact that the recurrent structure requires time both for training and inference. A faster solution can be represented by the Transformers architecture, which allows parallelizing the training procedures.

4.1.3 Transformer based models

In [53] the authors propose the employment of a transformer-based architecture for the punctuation restoration task.

The architecture proposed has two softmax layers: one for determining the word probabilities and one for the label probabilities (that determines the punctuation marks).

The model is trained in order to maximize the probabilities of observing the target label sequence and minimizing the words error rate, meaning:

$$\arg\max_{\theta} \frac{1}{N} \sum_{n=1}^{N} \Big(\log_{\mathbb{P}_\theta}(\boldsymbol{y}_n|\boldsymbol{x}_n) + \alpha \log_{\mathbb{P}'_\theta}(\boldsymbol{z}_n|\boldsymbol{x}_n) \Big) \tag{4.11}$$

For the punctuation prediction task, the second softmax would be sufficient but it is easy to consider that the class “no punctuation” is much more frequent than each symbol separately. The usage of the words-sequence softmax is justified by the fact that this way the model is able to treat each word and punctuation

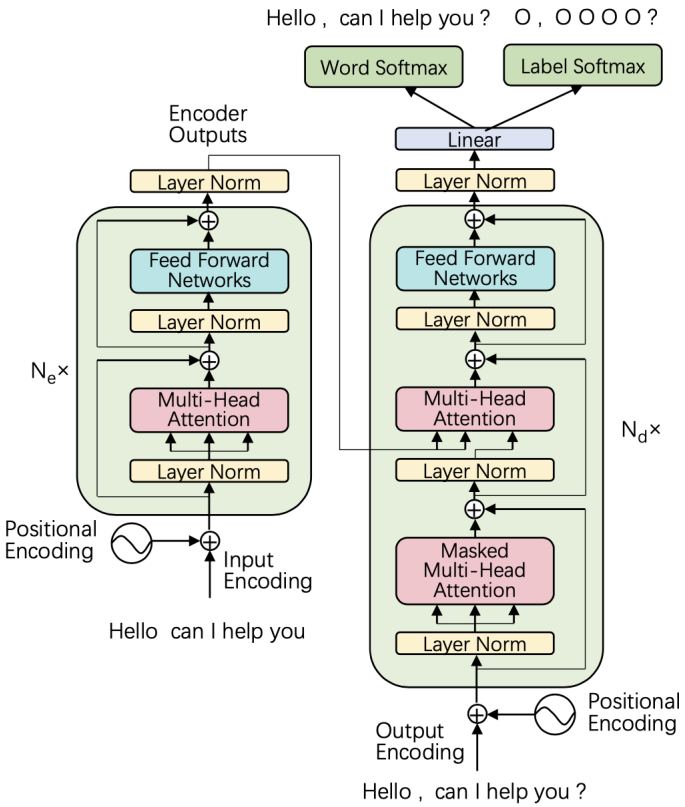

Figure 4.4: Structure of the transformer model presented in [53] for punctuation restoration.

mark equally, limiting the effect of class unbalancing. In the decoder, if the most probable symbol of the label softmax corresponds to the class *no punctuation* the next word is feed into the network, otherwise the corresponding punctuation mark.

In [54] the authors compare different transformer models and propose to add a bidirectional LSTM on top of the pretrained transformer network, and an augmentation strategy to improve the performance of the models: the transcripts are typically generated by an ASR system that can be affected by four kinds of errors (insertion, deletion, substitution, deletion), while models are trained with the correct text. The augmentation strategy should be able to simulate such errors and dynamically augment the number of sequences on the fly.

The best performing models also available in the library derives from BERT family, the RoBERTa model.

RoBERTa model

The basic architecture is proposed in [55] where the authors propose some modifications in BERT architecture and training procedure in order to reach better results. More in detail, firstly they expand BERT architecture.

In the second instance, starting from the assumption that BERT-based models rely on large quantity of data, they propose a *dynamic masking* procedure in order to increase the variability and the robustness of the model.

The original BERT implementation performed masking once during data preprocessing, resulting in a single *static masking* and the same input is repeated n time in the dataset, each time with a different mask. In the *dynamic masking* the authors generate the masking pattern every time the sequence is feed to the model.

The MLM training phase becomes longer, so the authors propose of removing the next sentence prediction objective as they assert that the removal of this task has a limited impact.

Moreover, they increased the batch size accepted by the architecture, and they noticed that by doing this, the model is capable of managing longer range dependencies.

4.2 Text summarization

Text summarization is the task of producing a new text starting from an initial one, which has the following properties:

- the final text length is shorter than the length of the initial text.
- the reduced text contains the same key informational elements that are present in the original text; furthermore, the content and meaning of the original text must be preserved.
- on the contrary, a summary does not include redundant information.

The automatic text summarization is able to produce fluent summaries without human intervention.

This task in practice is very challenging, for different reasons according to [56] and [57]:

- unlike the images, for which it is easier to define a fixed and predefined size, input text length may be different; a text is typically composed by different sentences and the number of sentences may vary from text to text, moreover, the length of each sentence varies as well; this variability requires models that are able to adapt according to this kind of input;
- Texts require an encoding/decoding procedures in order to transform char-based/string-based sequences into real number sequences;
- In some cases, to understand the meaning of a text, it is demanded to process it entirely. And from a computational standpoint this process may require nontrivial efforts;
- Texts may be subjected to variability due to the grammar of a text: the mutation of a language is a process that occurs over time and it is subject to cultural, geographical and economic factors; the most evident example may be the English language, which is spread in more than 10 countries as mother tongue and diffuses in the rest of the world as the language of business; the process of diffusion has resulted into the diversification and the pluralization of the English;

It is important to remark that the analyzed dataset contains speech transcriptions and not plain text. The speech transcript summarization task is not been widely explored in literature and the existing models make use of text summarization methods and try to adopt it in the case of speech.

Automatic text summarization can be carried out via *extraction*, meaning selecting the most representative sentences in a text and concatenating them for

a summary, or by *abstraction*, which involves generating novel sentences starting from the vocabulary of the input text.

4.2.1 The extractive summarization

This approach aims to build the text summary by selecting the most informative sentences in the original text and join them together to form a coherent summary. More deeply, this approach firstly constructs a representation in floating-point vectors of the input text. Then, assigns a score to each sentence according to the text representation. This score is proportional to the probability that the corresponding sentence will be selected as relevant and will compose the summary. Consequently, the sentence is ordered according to this score and the top k most relevant sentences are selected to be part of the summary. This kind of approach allows respecting grammar rules and the properties of languages as the summary is built according to the sentence present in the original text.

Statistical and LSA-based methods

These methods are based on the following pipeline: first define the base unit (the components that will form the summary), then proceed with the extraction of the score for each unit and sort them according to this metric, and select the best subset.

Kupiec et Al in [58] present an algorithm that, by the mean of a Bayesian classifier, is able to determine if the sentence is relevant and has to be included in the summary.

After having defined the set of k features that describe each sentence (denoted as F_j for $j = 1, ..k$), and the sentence as the base unit, they compute a probability score that represents the probability that the sentence s will be included in the summary S:

$$\mathbb{P}[s \in S|F_1, F_2, ..., F_k] = \frac{\prod_{j=1}^{k} \mathbb{P}[F_j|s \in S]\mathbb{P}[s \in S]}{\prod_{j=1}^{k} \mathbb{P}[F_j]} \tag{4.12}$$

In (4.12) the probability derives from the Bayes rule and additionally the authors assume the independence of the features. Therefore the algorithm chooses the set of sentences S' that maximize this probability, by estimating the terms in the formula directly from the training set.

In [59] the authors propose an unsupervised algorithm to extract semantic properties of a text by the means of statistical and algebra analysis. This method comes from the idea that the aggregate of all the word contexts is able to determine the similarity of semantic among words or set of words [60]. Latent Semantic

Analysis (LSA) is an algebric technique of analyzing relationships between a set of documents and terms they contain by producing a set of concepts related to those.

The information retrieved by this algorithm concerns the relationships among words as they appear in documents, and highlights, according to the frequency, the ones that are semantically related. This algorithm is composed by different steps:

1. Creation of the documents-terms matrix A: given a set of documents $\mathcal{D}$ of cardinality m, each document contains a set of words $d_i = w_1, w_2, ..., w_{M_i}$ of length M_i. Given m documents and n terms in the document vocabulary, a $[n \times m]$ matrix is composed by the sentences in the columns and the words in the rows; for each pair of sentence-word a scalar value is inserted according to the presence of this word in the corresponding sentence. In the documents-terms matrix each sentence is represented by a set of weights that can be obtained via, different frequency-based weighing schema such as:

   - Binary representation of the word: each entry of the matrix A can assume only two values $a_{i,j} = \{0, 1\}$ where 1 indicates that the presence of the term i in the sentence j, 0 otherwise;
   - Frequency of the word in that sentence: each entry of the matrix A represents the number of times in which the term i is contained in the sentence j, denoted as $TF(i, j)$;
   - TF-IDF (Term Frequency-Inverse Document Frequency): that evaluates the importance of words in a specific sentence over its presence in the complete document; in other words, this term shows how much the words characterize a sentence. This metrics merges two terms: the term frequency as previously defined and the inverse document frequency, which measure the importance of terms by taking into account the rareness. More formally the IDF(i) term is given by:

     $$\text{IDF}(i) = \log\left(\frac{m}{m(i)}\right) \tag{4.13}$$

     where m is the total number of documents and $m(i)$ represents the number of documents in which the term i appears. The TF-IDF(i, j) is given by:

     $$\text{TF-IDF}(i, j) = \text{TF}(i, j) * \text{IDF}(i) \tag{4.14}$$

2. Decomposition of the documents-terms matrix A via Singular Value Decomposition; the matrix A is factorized as follows:

   $$A = U\Sigma V^T \tag{4.15}$$

   where U is a column orthonormal [words $\times$ concepts] matrix $[n \times m]$ dimensions, V, the orthonormal matrix that represents the sentence $\times$ concepts matrix

with dimension $[m \times m]$. Once the matrix A is factorized in these three matrices, the first k columns of U ad V are chosen, giving:

$$A_k = U_k \Sigma_k V_k^T \tag{4.16}$$

where A_k represents the approximation of A at rank k, and so selecting the fist k concepts. The analogy with the concepts derives from the following reasoning: as previously mentioned, the matrix A is computed by taking into account some frequency-based weighting schema. The structure of A is then divided into three matrices which are capable of capturing the relatedness of some terms in A. The co-occurrence of some terms is, in most cases, driven by the context in which the terms appear. By projecting A in a subspace of dimension k, where the basis represents concepts, documents that contain similar words will be projected closely into a certain region of the k-dimensional space. This way, the Singular Value Decomposition is capable of capturing the latent semantic structure of the original document. Moreover, if a series of words is more likely than others, a singular value included in Σ matrix will take into account of that. The magnitude of that singular value is representative of the importance of the corresponding pattern. If each words pattern describes a concept, each singular value defines how much the corresponding concepts is relevant in the complete document. The more a document is similar to a certain pattern (where the similarity is given by the common terms), the higher will be a certain value in its singular vector representation in correspondence of the concept defined by the pattern.

3. Sentence selection: according to [61] the sentence selection can be performed with different techniques. [62] suggested to consider the right singular vectors matrix, that returns the relationship among concepts and sentences: for each concept, the most important sentence is selected (the one associated with the highest score). But this approach has two drawbacks: firstly it forces to select one sentence for each concept, regardless of the importance of the concepts itself. Thus, the length of the summary will be proportional to the number of concepts. The authors in [63] suggest other methods, which employ the singular value matrix in order to select the sentence for which the concept is lower than a threshold and multiply them to its corresponding singular value, to emphasize the most important concepts. Mathematically, given the document i, the algorithm orders the documents according to a score computed as follows:

$$s_i = \sum_{j=1}^{k} v_{i,j}^2 \cdot \sigma_j^2 \tag{4.17}$$

where s_i is the saliency score: the importance of a sentence is both determined by the relatedness of it with a series of k concepts and the importance of these k

concepts themselves. Authors in [64] suggest introducing a preprocessing phase before the sentence selection step. The preprocessing consists in computing the average sentence score as the average over the row of V^T and zeroing all the entries whose value is below its corresponding sentence score.

Graph-based methods: TextRank

The algorithm in [65], called *TextRank*, is a graph-based algorithm that aims to order the sentence in a text according to their importance. It took inspiration from Google's PageRank [66], developed for ordering web pages in a Google search. It is based on the assumption that the nodes that are connected to many other nodes are more likely to carry relevant information about the entire corpus, so the sentence selection process is driven by a graph index that is produced by an established ranking algorithm. In the *TextRank* graph, the node represents a unit, while the edge represents a semantic relationship that occurs between the two units in the edge. In the context of summarization, the unit is given by a sentence and the connections will be a metric of similarity between pairs of sentences.

This algorithm is based on the concept of recommendation. If two nodes in a graph are connected, that means that they can be related to each other, and this recommendation (the similarity measure) determines the importance: the more a sentence is linked to other sentences, the more is recommended and probably will assume more importance. The recommendation degree is given by a set of weights, according to the importance returned by all the nodes that send a vote.

More formally, give a graph $G = (V, E)$ where V are the set of vertices and E is the set of edges, if the set of all incoming edges of a node V_i is denoted as $\text{IN}(V_i)$ and the set of all outgoing edges as $\text{OUT}(V_i)$, the original formulation of the algorithm defines the vertex score as:

$$S(V_i) = (1-d) + d * \sum_{j \in \text{IN}(V_i)} \frac{1}{|\text{OUT}(V_i)|} S(V_j) \tag{4.18}$$

That, in the TextRank algorithm is equivalent to:

$$\text{TR}(s_i) = \frac{(1-d)}{N} + d \sum_{s_j \in \text{IN}(V_i)} \frac{w_{ij}\text{TR}(s_j)}{\sum_{s_k \in \text{OUT}(V_j)} w_{ik}} \tag{4.19}$$

Where $\text{IN}(V_i)$ is the set of incoming edges of the vertex V_i associated to the sentence s_i, $\text{OUT}(V_j)$ is the set of outgoing edges of the vertex V_j associated to the sentence s_j, w_{ij} is a similarity score between the sentences s_i and s_j.

So, the score of each node is due to:

- the damping factor d, which assumes values between 0 and 1, representing the probability of jumping from a node to another random vertex in the graph.

In PageRank it is defined as the probability that a "random surfer" (i.e. a user) will get bored and request another random page;

- the mean score of all the connected nodes where the score is represented by a measure of similarity among sentences.

In the context of NLP there are several similarity metrics according to [67], that can be divided into:

- String-based similarity metrics: which calculate the similarity at characters or terms level;
- Corpus-based similarity metrics: in which the similarity is given by the amount of information gained from the entire corpora;
- Knowledge-base similarity metrics: that make use of semantic networks.

For TextRank algorithm, the similarity is given by the overlap of two sentences, which can be determined simply as the number of common tokens between the two sentences, or given by the cosine distance:

$$w_{ij} = \frac{X_i^T ... X_j}{||X_i^T|| ||X_j||} = \frac{\sum_{k=1}^{m} X_{ik}^T \sum_{k=1}^{m} X_{jk}^T}{\sqrt{\sum_{k=1}^{m} (X_{ik}^T)^2} \sqrt{\sum_{k=1}^{m} (X_{jk}^T)^2}} \tag{4.20}$$

The score update proceeds iteratively until it converges below a certain threshold. Then the top-ranked sentences are selected for inclusion in the summary.

To sum up, the main steps of the algorithm for text summarization are the following:

- Construction of a Bag-of-Words-like representation of the corpus X (sentences in column, and terms in row).
- Construction of the pairwise similarity matrix, in which each entry i and j represents the similarity score between two sentences i and j.
- Generation of the undirected graph where each node is a sentence in the corpus, and each edge has a weight that corresponds to the similarity score between the two sentences;
- Assignment of a score of each sentence according to 4.19;
- Sentence ranking building, according to the score previously computed;
- Selection of the top-k sentences.

Other techniques

There are several kinds of techniques different from the previously mentioned, here briefly described for the sake of completeness.

Cluster-based techniques

Cluster-based techniques compare different sources and extract the most relevant part of them as an outcome of this comparison [68]. The algorithm divides the sentences coming from different documents into several clusters, and after having identified the top-k most populated clusters, the algorithm extracts a set of sentences among those selected. The idea behind this is that the most populated clusters will be the ones associated with the most important topics in the set of documents, furthermore, the sentence extraction is applied to those. Authors in [15] proposed to use a pre-trained language model to produce the embedding representation of each sentence, and then apply an unsupervised clustering algorithm to perform the selection. After computing the cluster-topic centroids, the algorithm computes the cosine distance among the learned representations of all the sentences and the centroids. Then it picks the most representative sentences, so the closest to the centroids.

Semantic-based techniques

The semantic-based approaches are based on the concept of *lexical chain* [69] and the employ of WordNet [70]. WordNet thesaurus is a vocabulary of synonyms that contains the information of relationships among terms. A *lexical chain*, as the name suggests, is a sequence of words that have a particular relationship to each others in the vocabulary used. The summarization process is divided into three steps:

- Segmentation of the original text;
- Construction of lexical chains, with the adoption of a scoring function for lexical chain evaluation; each chain has a score that is proportional to the number of terms contained and the homogeneity of the chain;
- Identification of the relevant sentences starting from the most significant chains: the strongest lexical chain will represent the most important concepts. Via heuristic algorithms, it is possible to choose the sentences that best approximate the concept extracted.

A drawback of this approach is that this method is not capable of controlling the level of details, and it has been proven that long sentences have a higher probability of being selected.

4.2.2 The abstractive summatization

Different from the extractive approach, the abstractive methods generate a completely new text, with sentences that are not directly picked from the original text. This task introduces a new challenge with respect to the previous extractive methods: the summary has to satisfy not only the requirement of containing all the information of the original text, but also the new sentences generated have to be correct from a grammatical and structural standpoint. For this reason, abstractive summarization requires the rephrasing of sentences in order to generate summaries that are linguistically fluent. Abstractive summarization methods, according to [71] can be divided into three categories:

- Structure-based approaches: by the means of pre-defined structures such as templates and ontology for information extraction.
- Semantic-based approaches: they create semantic graphs for defining the relationships between words and sentences.
- Deep learning-based approaches: these methods make use of deep learning models.

For the purpose of this thesis, only the deep learning-based approaches are described. Among those, it is possible to find methods that use Recurrent Neural Networks or Transformer architectures.

Recurrent models

As presented in [72], the authors create a novel method that involves local attention model (originally presented in [52]) in order to build abstractive summarization, with an approach more data-driven. They highlight two important aspects: firstly, the output sequence length is decided a priori and remains fixed, and secondly, the abstractive summarization methods allow more freedom in the generation process and let the model be trained with a wider range of training data.

The model proposed is composed by:

- An encoder model: starting from the input text, they are able to return a vector that represents the input and its context. They propose several kinds of encoders such as the simplest Bag-of-Word encoder, the Convolutional encoders (which are able to consider interactions between neighbor words by the means of standard time-delay neural network convolutions) and the most complex Attention-based encoders, similar to the one described in Section *The attention mechanism and the late fusion approach* for the punctuation restoration task. And this last typology was proven to be more effective. The key idea is to define an encoder capable of evaluating the input word importance with

respect to a specific word in the output, in this case, represented by the text summarized. The input representation is based on its context.

- The generation model: the algorithm that aims to build the summary.

Given the training data $(\boldsymbol{x^{(1)}}, \boldsymbol{y^{(1)}}), ..., (\boldsymbol{x^{(J)}}, \boldsymbol{y^{(J)}})$ consisting in tuples of input text and summaries, the model proposed is trained to minimize the negative log-likelihood of each summary, i.e:

$$\text{Loss}(\theta) = -\sum_{j=1}^{J} \log \mathbb{P}[\boldsymbol{y}^{(j)}|\boldsymbol{x}^{(j)}; \theta] = -\sum_{j=1}^{J}\sum_{i=1}^{N-1} \log \mathbb{P}[\boldsymbol{y}_{i+1}^{(j)}|\boldsymbol{x}^{(j)}, \boldsymbol{y}_c; \theta] \tag{4.21}$$

During inference, the model chooses the best summary as the set of tokens that maximizes:

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y}\in\mathcal{Y}} \sum_{i=0}^{N-1} g(\boldsymbol{y}_{i+1}, \boldsymbol{x}, \boldsymbol{y}_c) \tag{4.22}$$

For computational reasons, the beam search is applied to the decoder for limiting the growth of the set of summary candidates.

Transformer-based models

Recent work leveraging pre-trained transformer-based sequence-to-sequence models and adapt them to abstractive summarization. The three most popular transformer architectures in this field are:

- BART [28]: a denoising autoencoder composed by a bidirectional encoder (based on BERT) over corrupted text and a left-to-right autoregressive decoder (based on GPT); a more accurate description is reported in Section *BART model*;
- PEGASUS [73]: this architecture is a large transformer-based encoder-decoder model trained on a massive text corpora; during pre-training, the important sentences are masked from an input document, and the model is trained to build this sentences starting from the remaining ones (Gap Sentences Generation);
- T-5 [74]: the original paper proposes a unified framework to combine all NLP problems into a single text-to-text format; the Text-To-Text Transfer Transformer uses the same model, loss function, and the same hyperparameters for various NLP task; T5 is an encoder-decoder model that can be trained in a supervized or unsupervized way.

The papers showed that the Transfer Learning-based model achieved considerable improvement for abstractive text summarization.

4.3 Summarization of learning data

According to [10] the volume of data in textual form in the educational context is very large, and exploring it can be extremely time-consuming.

Learning From Summaries (LFS) [75] is a new data mining approach that exploits document summarization techniques in order to support learning activities. In particular, the pipeline requires a first step of document collection and annotation, where users annotate documents with notes or highlights. Then the documents are preprocessed and only textual portions of the documents are considered in the summarization stage, which is carried out through the Itemset-based Summarizer (ItemSum) [76].

The paper in [10] proposes a new methodology to recommend summaries of potentially large teaching documents.

The methodology proposed is the following:

- *Formative assessment*: multiple-choice questions are employed to assess the students' level of understanding;
- *Text preparation*: preprocessing phase of the results of multiple-choice tests;
- *Summarization*: reduction of the teaching material according to the topic;
- *Recommendation*: suggestion of the summaries to students who have not passed some particular tests.

For the summarization step, the authors proposed the usage of the Weighted Itemset-based Summarizer (MWISum) [76].

Authors of [11] review some implementations of automatic text summarization in the learning context from the year 2010 to 2020, mostly based on extractive summarization.

Another study focuses on mobile learning [77] and investigates the effectiveness of automatic text summarization used in the mobile learning context.

In [15] the authors propose a topic-based summarization from online discussion, leveraging on the threads related to the topics discussed. According to the authors, discussions (even when in an online version) lead to a better learning experience if adequately moderated. After a data processing stage, in which the textual data is lowercased, lemmatized and stopwords are removed, the authors propose different algorithms for the *Topic Extraction* phase.

[16] proposes a novel method of summarizing lecture slides. The goal of this work to enhance preview efficiency and improve students' understanding of the content. More in detail, they combine three kinds of features to compute the importance of a document: the first feature is determined by content volume, estimated by counting the number of foreground pixels, a second feature is the

inter-frame distance, which denotes changes between successive pages. The third feature is given by the TF-IDF of all the words that the document contains.

The closest works to the proposed problem are the following. In [17] the authors propose a method for producing extractive summaries from the source subtitles. They propose the usage of a pipeline that consists in:

1. Input text pre-processing with the classic NLP techniques, such as stop word removal, lemmatization;
2. Features selection (at word and sentence level);
3. Conversion to TF-IDF and normalization;
4. Sentence selection: the authors propose to verify if the TF-IDF of a sentence is greater than the mean TF-IDF present in the text.

Even the authors in [12] proposed an unsupervised extractive summarization model, in this case based on BERT architecture. In particular, the transformer model is pre-trained and then is employed to generate the embedding of the sentence. Then K-Means and Gaussian Mixture Models are used for clustering, utilizing the Sci-kit Learn library's implementation [1]. Once the clusters are computed, sentences closest to the centroids are selected for the final summary.

To the best of our knowledge, no work in the existing literature proposes to apply abstractive summarization techniques for video lecture summarization.

[1] Available at: `https://scikit-learn.org/stable/`

4.4 Transfer learning and domain adaptation

4.4.1 Formal definition of transfer learning

The availability of large-scale dataset corpora and the adoption of large pretrained language models allow advances in different NLP tasks, including summarization. Creating large datasets for a new domain, however, is often infeasible and highly costly. Thus, the ability to transfer knowledge from large pretrained models to new domains with little or no in-domain data is necessary, especially when such models should be adopted in real-life applications.

But a common assumption of machine learning models is that the training and test data are drawn from the same feature space and the same distribution. So, when the distribution changes, most of the models need to be revised using newly collected training data, similar in distribution to the one provided in the test set. Interesting studies [78] proved that larger models do not necessarily generalize better out-of-distribution, while in general pretraining on more diverse datasets can improve robustness.

Transfer learning [79] is the technique of knowledge transfer or transfer learning between task domains. The author in [80] reviews a 1970s research and provides the mathematical background of transfer learning. It concludes with a discussion on some computer vision applications.

If a domain is denoted as $\mathcal{D} = \{\mathcal{X}, P(X)\}$, where $\mathcal{X}$ is the feature space and $P(X)$ is the marginal probability distribution over that feature space and a task is represented by $\mathcal{T} = \{\mathcal{Y}, P(Y|X)\}$, where $\mathcal{Y}$ is the label space and $P(Y|X)$ is the target posterior probability distribution, a formal definition of transfer learning can be the following: defining a source domain $\mathcal{D}_\mathcal{S}$ and its corresponding source task $\mathcal{T}_\mathcal{S}$, and the target domain as $\mathcal{D}_\mathcal{T}$ with its target task $\mathcal{T}_\mathcal{T}$, the objective of transfer learning is to learn the target conditional probability distribution $P(\mathcal{Y}_\mathcal{T}|\mathcal{X}_\mathcal{T})$ (where $\mathcal{Y}_\mathcal{T}$ and $\mathcal{X}_\mathcal{T}$ are the training data of the target domain) in $\mathcal{D}_\mathcal{T}$ with the information gained from $\mathcal{D}_\mathcal{S}$ and $\mathcal{T}_\mathcal{S}$ where $\mathcal{D}_\mathcal{S} \neq \mathcal{D}_\mathcal{T}$ or $\mathcal{T}_\mathcal{S} \neq \mathcal{T}_\mathcal{T}$.

4.4.2 Transfer learning in NLP applications

In NLP, many tasks share common knowledge about language and according to Professor Andrew Ng "transfer learning will be [...] the next driver of ML commercial success" [2].

In [81] the authors define *linguistic intelligence* as the ability to reuse linguistic knowledge for a new task, in contraposition of *catastrophic forgetting*, the phenomenon for which a model loses any information previously acquired as soon as it

[2] Andrew Ng on transfer learning at NIPS 2016

approaches a new task. The authors discover that although SOTA language models have a large number of parameters and are subject to long unsupervised training, fine-tuning these architectures on a downstream task still requires a consistent number of training examples. In their experiments, the authors investigate the importance of selecting tasks to pre-train on and found that training curriculum (i.e. the dataset domains used during pre-training) can have a considerable effect on model performance on a new dataset.

In NLP applications there are different kinds of transfer learning:

- Domain adaptation: in this case the source and the target task coincide but labeled data are provided only in the source domain, which differs from the target one (transductive transfer learning).
- Cross-lingual learning: again source and target task are the same and only the source dataset is labeled, and the source and the target languages are different (transductive transfer learning).
- Multi-task learning: in this case the source and the target task are different (inductive transfer learning);
- Sequential transfer learning: similar to multi-task learning but in this case the tasks are learned sequentially (again part of inductive transfer learning).

As said in the previous sections, given the lack of educational datasets and the existence of models in literature trained on large datasets whose performance is noticeable, the thesis will explore the capability of transferring knowledge to the educational domain.

In NLP, domain means a type of corpus, with a certain topic, style, genre, or linguistic register [82]. According to the paper, a corpus is a set of instances drawn from an unknown high-dimensional distribution, called *variety space*, whose dimensions (or *latent factors*) define language aspects. They can be the genre, the sub-domain, can involve socio-demographic aspects, style.

A more formal definition starting from the previous definition of transfer learning of domain adaptation can be: in domain adaptation the source and target tasks, $\mathcal{T}_\mathcal{S}$ and $\mathcal{T}_\mathcal{T}$, remain the same, but the source and target domains $\mathcal{D}_\mathcal{S}$ and $\mathcal{D}_\mathcal{T}$ differ in their underlying probability distributions. Given two distributions $\mathcal{P}_\mathcal{S}(X, Y)$ and $\mathcal{P}_\mathcal{T}(X, Y)$, domain adaptation typically addresses the shift in marginal distribution $\mathcal{P}_\mathcal{S}(X) \neq \mathcal{P}_\mathcal{T}(X)$, also known as *covariate shift*.

The paper presents different methods to transfer knowledge and divides them into *model-centric*, *data-centric* and *hybrid* approaches. Model-centric approaches redesign parts of the model. They include a redesign of the feature space via feature augmentation and feature generalization methods, a revision of the loss function or regularization, that for example employ domain adversaries.

Regarding this work, the thesis focuses on analyzing the impact of domain shifts from meeting to educational, and explore the case in which no examples of the educational dataset are given to the model, similarly as in the zero-shot learning [83] [84] (with the only exception that the task is always summarization), in order to measure the domain adaptation capabilities.

Chapter 5

Proposed methods

This section is devoted to explaining the proposed methods for summarizing video lecture transcripts. In particular, in the following sections, two methods are proposed:

- A modified version of a Microsoft model, originally designed for meeting summarization. This model exhibits poor domain adaptation capability and the model's computational complexity quadratically depends on the input text size: this is a nontrivial limitation in the e-learning context, where the transcripts usually have a length of about 8000 of tokens.
- A novel method that merges extractive and abstractive summarization: such method consists in a first stage of unsupervised extractive summarization algorithms to filter out useless sentences and to generate a short text that respects the limit of the abstractive summarization model; the extractive summarization stage does not require the input text to have a fixed length.

The chapter ends with a comparison of the complexity of the two proposed models, and a summary of the improvements introduced with the novel approach. In the video lectures context, the proposed method achieves better performance than Microsoft's architecture in terms of ROUGE-n scores and BERTScore; for a more detailed description see Section *Comparison of the two models*.

5.1 A hierarchical transformer model for meeting summarization

In [85] authors propose a Transformer architecture that is capable of performing meeting summarization. The datasets under analysis are the AMI and the ICSI dataset. As highlighted in the referenced paper, the meeting transcripts are typically longer than plain text: thus, performing summarization on them becomes extremely challenging. For example, in CNN/Daily Mail dataset [34], there are on average 781 tokens per article and 56 tokens per summary, while AMI meeting corpus contains meetings with 4,757 tokens per transcript and 322 tokens per summary. Moreover, a meeting is carried out between multiple participants that contribute to the meeting with their different roles and standpoints. To address these issues the authors propose a Hierarchical Meeting summarization Network, an encoder-decoder transformer model that leverages on the information of roles to generate abstract summaries. To tackle the problem of meeting length, the authors propose a hierarchical architecture composed of three layers:

1. A word-level encoder (WT): it processes one token at a time (including its corresponding POS and NER), and produces a token representation:

$$\mathrm{WT}(\{x_{i0}, ..., x_{iL_i}\}) = \{x_{i0}^W, ..., x_{iL_i}^W\} \tag{5.1}$$

   where $x_{i0}, ..., x_{iL_i}$ are the input vectors (token + POS + NER) of the sentence i with length L_i. Only the first output of the word-level encoder is used in the subsequent turn-level encoder, the one that corresponds to the tag that indicates the beginning of the sequence, i.e. x_{j0}^W .

2. A turn-level encoder (TT): takes as input the first representation of the word-level transformer and concatenates it with the role vector of the speaker for this turn:

$$\mathrm{TT}(\{[x_{10}^W; p_1], ..., [x_{m0}^W; p_m]\}) = \{x_1^T, ..., x_m^T\} \tag{5.2}$$

   where $x_{10}^W, ..., x_{m0}^W$ are the output representations from the word-level encoder and $p_1, ..., p_m$ are the role vector representations. Afterwards, the encoder processes the information of all the turns in a meeting; this portion of the model is able to capture the relevant information coming at word-level and to condense it in a unique vector.

3. The decoder: this model is capable of generating a new summary using a lower triangular mask to prevent the model to look at future tokens and cross attention with word-level and turn-level transformer. During training, the authors use the teacher-forcing approach, while at inference time the decoder exploits its autoregressive nature. In order to improve the results, the authors use beam search to select the best sequence of candidates.

A schema of the described architecture is depicted in Figure 5.1.

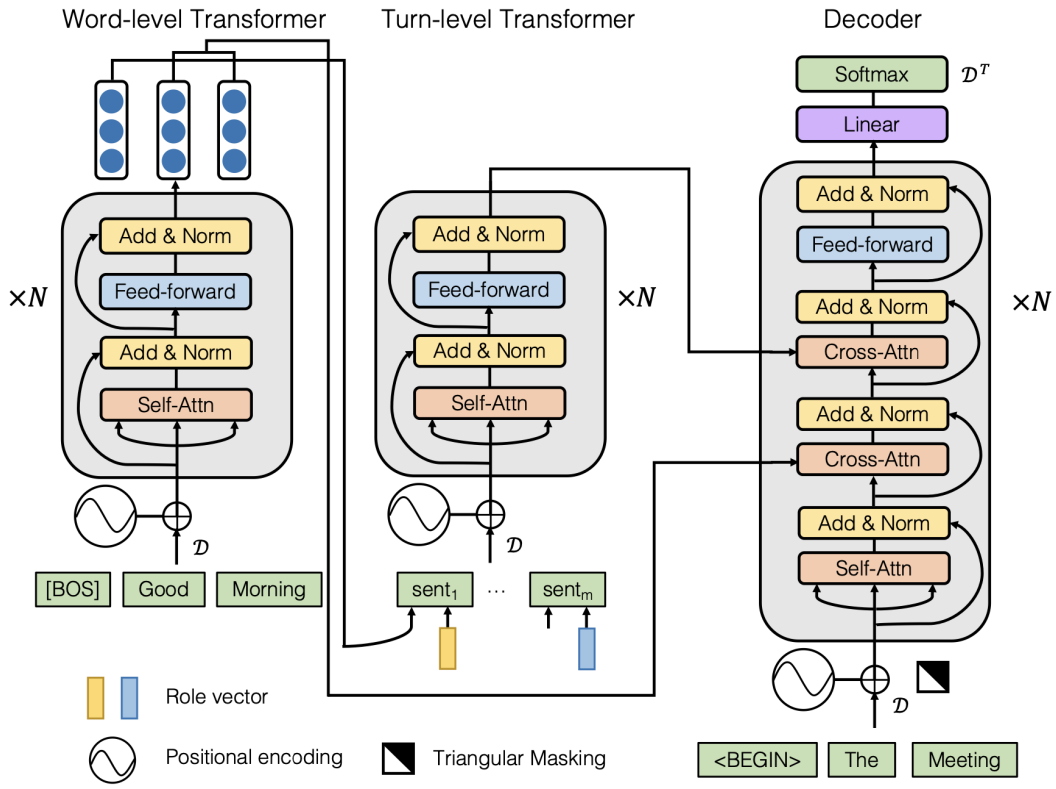

Figure 5.1: The hierarchical structure for meeting summarization presented in [85] based on the transformer architecture.

Given the limited amount of training data for meeting summarization, before finetuning the model on the AMI and the ICSI dataset, the authors pre-train it on the CNN dataset. In particular, they concatenate a set of news articles into a series of people meeting, and treat each sentence as a single turn.

5.2 A speaker-aware version of HMNet with topics

For Microsoft's architecture, the authors provide two finetuned versions, one obtained with the AMI and one obtained with the ICSI dataset. The results obtained on the validation and test set with the complete architecture are displayed in Table 5.1.

Results of HMNet on AMI and ICSI dataset			
	ROUGE-1	ROUGE-2	ROUGE-L
AMI	0.530 (0.495)	0.186 (0.170)	0.493 (0.489)
ICSI	0.463 (0.490)	0.106 (0.113)	0.474 (0.480)

Table 5.1: The table above summarizes the results on the test and development set (the results of the development set are within brackets) obtained by the authors of HMNet with the original configuration on the AMI and the ICSI datsets.

Indeed, the results highlighted in the paper are derived from an architecture that requires some additional information about the transcriptions. Recalling what is stated in Section *The EduSum video lectures dataset*, video lecture transcripts differ from the AMI transcripts in three aspects:

1. The gold summaries in EduSum are shorter than the ones in AMI and ICSI;

2. The absence of multiple speakers and the unicity of the role: according to HMNet paper, the ROUGE-1 score drops by 5.2 points on AMI and by 2.3 points on ICSI when the role vector is removed.

3. Possible errors in punctuation: a video lecture transcription may contain punctuation errors, that, as previously described in Section *The Politecnico video lectures dataset*, is inserted by a pre-trained model.

In order to bring the architecture closer to the case under analysis, two modifications have been introduced. Firstly, the information of the speaker is replaced with the information of the concept: each sentence is associated with a topic, or concept, which is extracted via Singular Value Decomposition. More specifically, starting from a Bag-of-word representation X of the meeting that reports documents along the columns and words in row, the matrix X is factorized in three matrices, U, S and V^T. This last matrix expresses the concepts' degree of representativeness of each sentence, i.e. how much information about a concept a sentence is able to capture. As described in the paper for the CNN/Daily Mail dataset, a fixed number of concepts N is established a priori and each document is associated to a

speaker-label according to the matrix V^T, considering the highest value among the concepts. For each column of V^T the concept associated with the highest value is picked among the N most important concepts. Secondly, the summary output lengths boundaries are reduced to 100 and 200 tokens [1].

[1] In the original configuration an output summary can reach 500 tokens.

5.3 Extractive-Abstractive BART-based model

It is known that for transformer-based architectures the memory and time required for full self-attention models grow with the input sequence length n with a quadratic rate $\mathcal{O}(n^2)$. This is a nontrivial limitation for the employment of BART model in the video lectures context, where the transcriptions' length is at least 8 times bigger than news articles' length and the BART model accepts a fixed number of input tokens to avoid memory issues.

For this reason, this work proposes a novel approach called *Extractive-Abstractive summarization*: it consists in the usage of an unsupervised summarization method to preprocess the input text, followed by the BART architecture, which aims to shorten the produced summaries even further and to generate their abstractive counterpart. The first stage, the extractive summarization, allows to filter out useless sentences and to respect the input constraints of BART. The second stage produce the abstractive summary.

More in detail, the extractive stage consists in the shortening the original input texts by means of an extractive summarization technique; several algorithms have been experimented with:

- Latent semantic analysis: the algorithm chooses the most representative sentences of the input dataset according to the right singular vectors matrix and the singular value matrix of the term-frequency matrix. The former indicates the importance of a sentence with respect to a series of topics k, the latter instead highlights the importance of each topic. The product of the two matrices returns a re-weighting of the sentence importance in the basis of topic relevance.

- TextRank algorithm: the algorithm picks the best sentence according to a ranking that orders the sentences according to their similarity to the rest of the text. The official implementation of this algorithm is provided by [86].

- Cluster-BERT: the pre-trained language model BERT is employed to extract an embedding representation of the input at sentence level, and then a clustering algorithm groups the sentences according to a similarity score with each centroid found; each centroid defines the topics of the input text. In particular, the K-Means algorithm has been employed for clustering and the similarity metric adopted is the cosine distance.

- Random choice: the sentences are randomly picked until the size constraint is met.

- Truncation: only the first tokens of each input text are considered.

- A mix of the previous techniques: motivated by the limited amount of data, during training more variability is artificially introduced by applying a random method from all those listed above.

A mathematical description of these algorithms is contained in section *The extractive summarization*.

The choice of these extractive summarization methods is motivated by the fact that extractive summaries are not available in many cases, while these algorithms are unsupervised, having no need for outer labelling. Moreover, these algorithms have no limits in terms of input size.

5.3.1 BART model

The state of the art in text summarization is represented by BART [28], a bidirectional denoising autoencoder for pretraining sequence-to-sequence models.

The BART architecture derives from two important models in the NLP field: BERT [27] and GPT [26]. A graphical representation of BERT and GPT training is depicted in Figure 5.2 and 5.3. The encoder part of BART is given by the BERT model, while GPT represents the decoder part of the BART architecture.

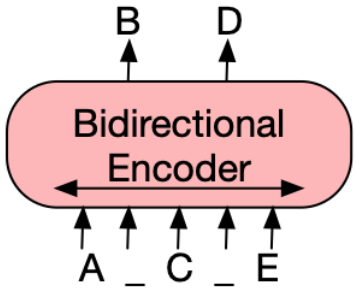

Figure 5.2: BERT architecture and random masking training schema from [28].

As previously stated, BART is trained via *sequence-to-sequence* training. A schema of this architecture is depicted in Figure 5.4. Starting from BERT and GPT building blocks, BART introduces some modifications:

- each layer of the decoder (GPT) performs cross-attention over the final hidden layer of the encoder (BERT);
- BART removes the last BERT feed-forward network before the word prediction layer;

The BART architecture is *pre-trained* by corrupting original documents and then forcing the model to find the best reconstruction of the original text. The corruption actions are the following:

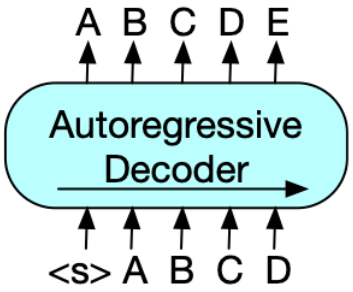

Figure 5.3: GPT architecture and training schema from [28].

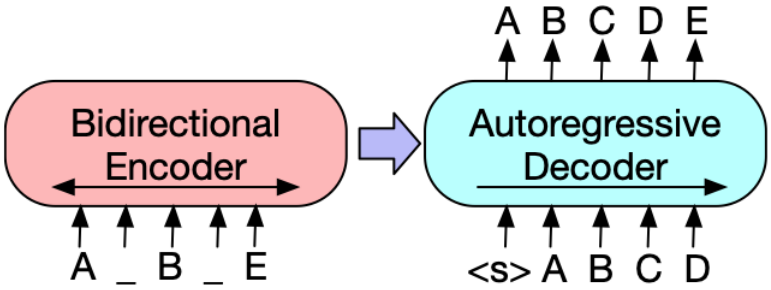

Figure 5.4: BART architecture and training schema from [28].

- Token masking: random tokens are masked with a token [MASK] and the model should determine the original token;
- Token deletion: random tokens are removed and the model has to decide which position are the missing inputs;
- Text infilling: a number n is sampled from a Poisson distribution, this number constitutes the length of word span replaced by the token [MASK]; when $n = 0$ it represents the insertion; the model is trained to determine the number of missing words;
- Sentence permutation: the order of each sentence is shuffled;
- Document rotation: the model is trained to detect the beginning of a sentence; a token is randomly chosen and the sentence is circulated from right to left up to that token.

In the *fine-tuning* stage, the model generates outputs in an autoregressive fashion. The main limitation of this model the input sequence length: BART is not capable of managing long streams of text, such as the ones under analysis. Thus, this model has been used to produce news or dialogue summaries.

BART models [28] are provided by Huggingface library [2].

The study in [87] highlights that the domain discrepancy between written language and spoken language may degrade the model performance. For this reason, the starting configuration chosen is a pre-trained BART model on the SAMSum dataset [35] for the summarization task. The usage of SAMSum pre-training configuration is motivated by the nature of video lecture transcripts, which is based on dialogues rather than newspaper articles. The first assumption is the following: a meeting corpus could be the closest choice for video lectures if a suitable training dataset is not available or if it contains a limited number of examples of that domain. After this preprocessing step, the deep architecture is fine-tuned in order to produce the abstractive version of the extractive summary with an even shorter length. This phase is particularly delicate, as language models have been shown to be unstable across different hyper-parameter settings. Although the pre-trained models are able to capture the semantic and the syntactic characteristic of a language, inadequate fine-tuning may produce undesired results and may generate overfitting. [88] is an interesting work done in order to understand the causes of this issue. The authors observe that pre-trained models change their behavior according to:

1. The weight initialization;
2. The training data order resulting from the dataset shuffling.

They assert that the large results' variability is particularly evident in smaller datasets. To mitigate this phenomenon the authors remark the importance of an early stopping exit strategy. Along with this procedure, in [89] the authors propose a regularization term in order to avoid the so-called *representational collapse*, i.e. the degradation of generalization capacity of pre-trained models during the fine-tuning phase. They show that it is possible to avoid this phenomenon by adding a regularization term on the output probability distribution in the form:

$$\mathrm{KL}(g \cdot f(x) || g \cdot f(x+z)) \tag{5.3}$$

where x is the input embedding learned representation, f and g are the functions that return respectively some pre-trained representation and the classification probability, and z is a sample drawn from a normal $z \sim \mathcal{N}(0, \sigma^2 I)$ or uniform distribution or $z \sim \mathcal{U}(-\sigma, \sigma)$.

By introducing a small variation on the input embedding vectors represented by z, it is possible to compute the distance, in terms of probability distribution, between the noised vector and the original vector: this distance penalizes the model if the two output probability distributions are far away from each other.

[2]The library is available at: https://huggingface.co

With this regularizer, the model should demonstrate its capability of generalization regardless of a small drift in the input representation. This regularization term is controlled by a hyperparameter λ.

Similar to [90], parts of the BART model are frozen and only the last layers of the encoder and decoder are fine-tuned, as well as the language model head.

Afterwards the model is fine-tuned on the AMI and the ICSI dataset. These dataset are chosen because meetings and video lectures share certain lexical features: they are both the results of spoken language carried out by one or more participants, and they both cover one or more topics. In the following chapter the fine-tuning choice is questioned and compared with pre-trained version of the model.

Algorithm 1 Extractive-abstractive test algorithm.

Data: Test set data: test_set
Input:

```
extractive()                    # Extractive method
PARAMS                          # Parameters for abstractive summarization
```

Output: The prediction summaries: results
Set:

```
model                           # Abstractive model trained
results ← empty list            # Output summaries
```

```
for epoch = {1, N_EPOCHS} do
    for (input_text, gold_standard) in test_set do
        extractive_text ← extractive(input_text)
        summary ← model.generate(extractive_text, **PARAMS)
        results.append(summary)
    end
end
```

Algorithm 2 Extractive-abstractive training algorithm.

Data: Training and validation: training_set, val_set
Input:

```
extractive()              # Extractive method
PARAMS                    # Parameters for abstractive summarization
```

Output: The model trained
Set:

```
model                     # abstractive model trained
loss_function()           # Criterion
optimizer(**PARAMS)       # Optimizer
scheduler(**PARAMS)       # Scheduler
```

```
for epoch = {1, N_EPOCHS} do
    for (input_text, gold_standard) in training_set do
        optimizer.zero_grad()
        extractive_text ← extractive(input_text)
        prediction ← model(extractive_text, **PARAMS)
        loss ← loss_function(gold_standard, prediction)
        loss.backward()
        scheduler.step()
    end
end
```

5.3.2 Considerations on complexity and efficiency

This paragraph is devoted to highlighting the benefits of the proposed method in terms of computational costs. HMNet architecture is made of more than 200M parameters, while BART does not exceed 140M. Using BART represents a twofold advantage: a smaller architecture has a smaller memory footprint, with an immediate 30% disk space saving. Moreover, a smaller model is also beneficial also in terms of computation, hence time and energy consumption.

This section presents some considerations on complexity, in order to underline the advantages of the presented model, not only in terms of numerical results. For the sake of simplicity, only LSA, TextRank [3] and the attention operation are compared, as they represent the main differences of the two approaches. Denoting with $\mathcal{V}$ the set of video lectures, with cardinality $|\mathcal{V}| = V$, with s_i the number of sentences in a video lecture v_i with a total number of tokens equal to n_i, and with p the vocabulary size, starting from a matrix $[p \times s_i]$ the time complexity of LSA for summarization is $\mathcal{O}(\min(ps_i^2, p^2 s_i))$ for SVD and $\mathcal{O}(s_i^2)$ for the sorting operation in the worst-case scenario.

For TextRank the time complexity according to [91] is $\mathcal{O}(k(m + s_i))$ where s_i is the number of nodes, so the number of sentences, m is the number of edges (equal to $\frac{(s_i(s_i-1))}{2}$ for a complete graph) and k is the number of iterations.

For any transformer architecture, the complexity associated with an attention layer is given by $\mathcal{O}(n_i d^2 + n_i^2 d + (dn_i d_1 + d_1 n_i d))$ where n_i is the number of tokens, d and d_1 are the dimensions of the feature spaces. The overall complexity can be broken up into several terms:

- the first term, $\mathcal{O}(n_i d^2)$, is given by the computation of the Query Q, the Key K and the Value V matrices, each of shape $[n_i \times d]$, obtained multiplying the input matrix with three learned matrices of dimension $[d \times d]$;
- the second term, $\mathcal{O}(n_i^2 d)$ is given by the attention mechanism, the product between the attention weights $\text{Softmax}(QK^T)/\sqrt{d}$ and the value matrix;
- the third term, $\mathcal{O}(dn_i d_1 + d_1 n_i d)$ is given the feed-forward stage, which consists in a two consecutive matrix multiplications of dimension $[d \times d_1]$ and $[d_1 \times d]$ with d_1 the dimension of the first matrix.

If the size of the vocabulary p and the dimensions d and d_1 are fixed, then when n_i grows, the complexity of attention models grows quadratically. For LSA and TextRank algorithm the complexity depends on s_i, for which it is reasonable

[3] As will be extensively described in the next chapters, these two algorithms allow to reach the highest scores, moreover, for the Extractive-Abstractive approach, this section takes into account LSA and TextRank only.

	Complexity	Approximation
LSA	$\mathcal{O}(\min(ps_i^2, p^2 s_i) + s_i^2)$	$\mathcal{O}(s_i^2)$
TextRank	$\mathcal{O}(k[\frac{s_i(s_i-1)}{2} + s_i])$	$\mathcal{O}(ks_i^2)$
Attention Layer	$\mathcal{O}(n_i d^2 + n_i^2 d + (dn_i d_1 + d_1 n_i d))$	$\mathcal{O}(n_i^2)$

Table 5.2: The table sums up the complexity of the algorithms considered.

assuming that $s_i < n_i$ holds [4]. It is also important to consider that Extractive-Abstractive model uses 10 attention layers, 5 in the encoder and 5 in the decoder. The input size $n_{i,\text{BART}}$ is fixed and determined by the chosen extractive algorithm. On the other side, HMNet has 18 attention layers in total (6 in each portion of the hierarchical structure) and the input size varies with the input meeting length, $n_{i,\text{HMNet}} = n_i$.

The complexity of Extractive-Abstractive model is given by:

$$c(\text{E-A}) \propto \mathcal{O}\Big(\max \Big[\min(ps_i^2, p^2 s_i) + s_i^2, k\Big[\frac{s_i(s_i-1)}{2} + s_i\Big]\Big]\Big) + 10 * \mathcal{O}(n_{i,\text{BART}} d^2 + n_{i,\text{BART}}^2 d + (dn_{i,\text{BART}} d_1 + d_1 n_{i,\text{BART}} d)) \quad (5.4)$$

While the complexity of the hierachical model is proportional to:

$$c(\text{HMNet}) \propto 18 * \mathcal{O}(n_i d^2 + n_i^2 d + (dn_i d_1 + d_1 n_{i,n_i} d)) \quad (5.5)$$

If the number of tokens n_i grows, both algorithm complexities grow, but Extractive-Abstractive architecture complexity increases with a lower rate, due to the fact that $s_i < n_i$ and the attention operations in HMNet are repeated many times.

The extractive algorithms before BART fix a limit in the complexity growth of the subsequent transformer model. For this reason, for a sufficiently large n_i, BART's complexity can be considered constant. The extractive algorithm complexities depend on s_i which, in the simplest case, can be considered a fraction of n_i (for example, it is possible to consider that a sentence, on average it is composed of 20 words, giving $s_i = n_i/20$).

For larger n_i, (at least $n_i > n_{i,\text{BART}}$), assuming $s_i = \frac{n_i}{k_0}$ where $k_0 > 1$, and given $\max(\mathcal{O}(s_i^2), \mathcal{O}(ks_i^2)) = \mathcal{O}(ks_i^2)$ the maximum complexity among LSA and TextRank, the complexity of Extractive-Abstractive model can be approximated

[4]The equality $s_i = n_i$ represents a very special case in which each sentence is represented by a single token, which it is ignored.

as follows:

$$c(\text{E-A}) \propto \mathcal{O}(ks_i^2) + k_1 \propto \mathcal{O}(\frac{kn_i^2}{k_0}) \tag{5.6}$$

while the complexity of Microsoft architecture is given by:

$$c(\text{HMNet}) \propto \mathcal{O}(18 * dn_i^2) \tag{5.7}$$

If $d > 0$ (reasonable to assume, as d is the dimension of the feature space in the attention mechanism) and $k_0 > 1$, it holds that $\mathcal{O}(\frac{kn_i^2}{k_0}) < \mathcal{O}(18 * dn_i^2)$ for $k < 18 * d * k_0$, giving $c(\text{E-to-A}) < c(\text{HMNet})$ for every $n_i > n_{i,\text{BART}}$.

In other words, the worse case is represented by TexRank + BART algorithm for which the complexity is lower than HMNet complexity with a controlled number of iterations k [5].

Empirical results of this proof are displayed in Figure 5.5: these experiments show the average time required for inference, run on a Tesla V-100, with 32 GB CUDA memory. In the case of Extractive-Abstractive approach, the time is comprehensive of the two steps.

However, if we consider LSA, it is possible to notice that the term of complexity $\mathcal{O}(s_i^2)$ is given by the sorting operation, which, according to the sorting algorithm employed, can be lowered to $\mathcal{O}(s_i \log(s_i))$.

For attention models several studies [92, 93, 94] propose variations on the attention mechanism to reduce its complexity. None of them, to the best of our knowledge, has been applied to spoken language yet and therefore are not taken into account.

In addition, it is important to point out that, the extractive summarization methods presented are currently considered as baselines in the literature, and there exist variants that improve their efficiency and results. Despite these simplifications, in most cases, the Extractive-Abstractive approach outperformes the hierarchical model in the video lectures domain, and it is likely that further enhancements can be obtained with more sophisticated methods upstream. In addition, Hugging Face library provides several versions of BART that can be replaced according to the memory constraints and with which it is possible to improve the results.

[5] Looking at the implementation of the Microsoft model, $d = 512$ and from the datasets considered $k_0 \sim 15$, therefore the upper bound of the number of iterations is more than 138000.

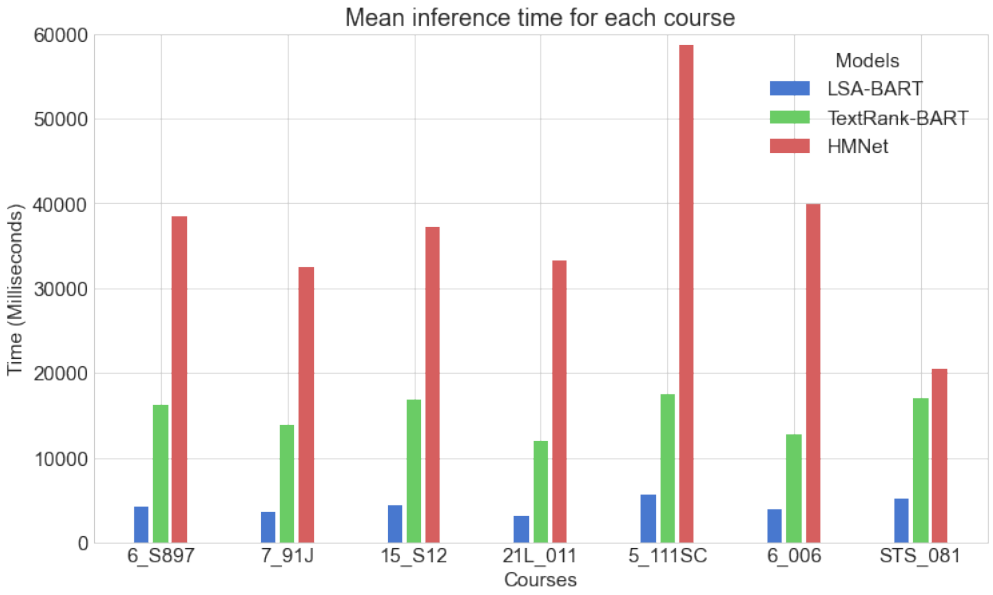

Figure 5.5: The mean time required for summary generation of HMNet and Extractive-Abstractive models for each course. This graph highlights the improvement gained with the proposed method in terms of time.

Chapter 6

Experiments

This section is dedicated to a discussion of the results obtained with the proposed methods. In particular, the first section *Experimental setup* defines the experimental configurations that lead to the results highlighted in section *Results on punctuation* and *Results on summarization*.

This last section analyses the results of the summarization restoration task. Several analyses have been conducted in order to determine if different preliminary assumptions are still valuable. As a baseline, the models are evaluated on the meetings datasets, and compared to each other.

The first assumption questioned is the usefulness of the punctuation restoration task. The following section will evaluate the impact that punctuation symbols have on the summaries generated by the different models. The second assumption to verify is the benefit of the meeting-to-video lecture transfer learning: the analysis calls into question the generalization capability of the model fine-tuned on meetings datasets with respect to the one only pretrained for general text summarization.

The evaluation always takes into account the different nature of the MIT courses which constitute the educational dataset. The Politecnico's gold summaries were not available: for this reason, only some examples of generated output are reported. The paragraph *Metrics* describes the metrics that have been used for the evaluation.

6.1 Experimental setup

6.1.1 Punctuation restoration experimental setup

This section describes the experimental setup adopted to restore the punctuation symbols.

A window-function approach is proposed in order to overcome the limitation of the punctuation restoration model, and the results are compared with the other methods.

In particular, two experiments are conducted. The first experiment is represented by a quantitative evaluation of results obtained through punctuation models, carried out with EduSum courses where punctuated transcripts were available.

Three attempts have been carried out in order to restore the punctuation of video lecture transcripts. In this case, no modifications have been developed as this part is considered not the core component of the thesis.

Firstly, the punctuation marks have been inserted with the Bidirectional RNN described in Section *Bidirectional Recurrent Neural Network*. In particular, the model that the authors of [48] call *T-BRNN* was employed. These results are compared with the ones obtained with the *FastPunct* library [1] which employs, as in the *Punctuator* architecture, a recurrent structure and the attention mechanism, which is based on LSTM units in the case of *FastPunct*.

Given the limitation of sequence length imposed by *FastPunc* model, a window-function is implemented as follows:

- The text is punctuated with the *Punctuator*, regardless of the poor results;
- According to the punctuation marks inserted, the punctuated text is divided into sub-units that contain at most N_{max} characters: the interruption of each unit is defined by the last punctuation mark before the $N_{max} - th$ character; the residual part of each sub-unit (i.e. the sequence of characters in between the last punctuation symbol and the N_{max}-th character) will compose the beginning of the subsequent sub-units;
- Each sub-unit is passed through the *FastPunc* model and the punctuated outputs are concatenated.

The third method proposed is the Transformer architecture of Alam et Al. described in section *Transformer based models*.

The second experiment is conducted with the Politecnico video lectures, where only a qualitative analysis is performed. In this case, the distribution of full stops, commas, and question marks is compared with that of the training set.

[1] https://pypi.org/project/fastpunct/

For all these experiments, computational resources were provided by HPC@POLITO| (`http://www.hpc.polito.it`). Experiments have been conducted with 1 NVidia Tesla V100 with 32 GB of CUDA memory 5120 CUDA cores.

6.1.2 Summarization experimental setup

As far as the proposed Extractive-Abstractive approach is concerned, the hyperparameters that lead to the performance described in the next paragraph are displayed in Table 6.1. In these experiments AdamW Optimizer [95], a variant of Adam [96], is employed. A more formal definition of these optimizers can be found in Appendix B.2.

The weight decay is set to 1e-4. The parameters for R3F regularizer are λ and σ (the variance of the noise, sampled from a normal distribution), respectively set to 1e-6 and 0.8 for all the configurations. The beam width for the output generation is always set to 5, with a repetition penalty equal to 2.

Hyperparameter for AMI dataset				
	Epochs	LR	Gamma	Step Size
LSA	10	1e-5	0.1	2
TextRank	2	1e-5	0.9	1
BERT	3	1e-5	0.9	-
Random	2	1e-6	-	-
Truncation	3	1e-5	-	-
Mix	2	1e-7	-	-
Hyperparameter for ICSI dataset				
	Epochs	LR	Gamma	Step Size
LSA	10	1e-5	0.1	2
TextRank	2	1e-5	0.9	1
BERT	2	1e-6	0.9	1
Random	2	1e-6	0.9	1
Truncation	2	1e-6	1e-4	1
Mix	3	1e-5	0.1	2

Table 6.1: The table reports the hyperparameters configuration used for finetuning BART on the AMI and the ICSI datasets.

HMNet's hyperparameters are mostly those of the original paper. The only difference consists in the parameters that control the beam search width and the maximum length for the output summaries, which are lowered to 3 and 250

respectively.

The experiments conducted are the following:

- Evaluation of the extractive summarization algorithms proposed for Extractive-Abstractive approach: among these only the two best performing models are kept for the following discussion;
- Evaluation of the proposed Extractive-Abstractive architecture in the meeting context after the finetuning stage;
- Evaluation of the proposed architectures with the educational dataset; in particular:
    1. Evalaution of the impact of punctuation restoration task in the summarization process both for Microsoft and the Extractive-Abstractive architecture;
    2. Evaluation of the usefulness of the meeting finetuning and the domain adaptation capabilities of the two models;
    3. Evaluation of the class subject impact on the summarization output.

For all these experiments, computational resources were provided by HPC@POLITO| (`http://www.hpc.polito.it`). The training of Extractive-Abstractive model and the inference with the Hierarchical architecture has been conducted with 1 NVidia Tesla V100 with 32 GB of CUDA memory 5120 CUDA cores.

The Hierarchical architecture training, according to [85], required 4 Tesla V-100, with a total 5120 CUDA core in total and 32 GB each. The authors provided the pre-trained model and fine-tuned versions of it.

6.2 Metrics

6.2.1 Punctuation evaluation metrics

Being a multiclass classification task, the punctuation restoration models are evaluated by means of the typical metrics used in classification. For this specific evaluation, it is important to take into account the imbalance of classes: the class *No punctuation* is much more frequent than the others. In case of class imbalance, accuracy may return an high score even if the minority class is not correctly classified.

Define the following:

- TP, true positive, the number of items correctly labeled as belonging to the positive class;
- TN, true negative, the number of items correctly labeled as belonging to the negative class;
- FP, false positive, the number of items wrongly labeled as belonging to the positive class;
- FN, false negative, the number of items wrongly labeled as belonging to the negative class.

One class c_i at a time is considered among *PERIOD*, *COMMA* and *QUESTION MARK*: the multiclass setting is turned into a binary classification problem. For each class c_i, the presence of the symbol after a word in the ground truth belonging to c_i is considered as assigning a positive label (1) to that words.

Then, the metrics adopted to evaluate the performances of a classifier are the following:

- Precision, or positive predictive value, is the number of TP divided by the total number of elements labelled with 1 (TP + FP), it highlights how valid the results are, i.e:
$$\mathrm{P} = \frac{TP}{TP + FP} \tag{6.1}$$
- Recall: is the number of TP divided by the total number of elements that actually belong to the positive class (TP + FN), it shows how complete the predictions are, formally:
$$\mathrm{R} = \frac{TP}{TP + FN} \tag{6.2}$$
- F-measure, that is the harmonic of precision and recall mean given by the following expression:
$$\text{F1-score} = 2 \cdot \frac{P * R}{P + R} \tag{6.3}$$

6.2.2 Summary evaluation metrics

In [97] the ROUGE score is presented as a set of metrics for evaluating automatic summarization. It compares a summary produced by the model against a set of ideal reference summaries. The acronym stands for *Recall-Oriented Understudy for Gisting Evaluation* and considers the number of overlapping units (n-gram, word sequences) among the prediction and the gold standard.

More formally, let R_S be the set of reference summaries, and let S be a summary that belong to this set. Then the metric is the following:

$$\text{ROUGE-N} = \frac{\sum_{S \in R_S} \sum_{gram_n \in S} count_{\text{match}}(gram_n)}{\sum_{S \in R_S} \sum_{gram_n \in S} count(gram_n)} \tag{6.4}$$

where n represents the length of the *n-gram* and $count_{\text{match}}(gram_n)$ is the number of times in which the n-gram $gram_n$ occurs in both gold standards and in the model output.

In the case in which multiple references are available, the final ROUGE score is given by the maximum among the pairwise ROUGE-N scores obtained with the model prediction against all the possible reference summaries, meaning:

$$\text{ROUGE-N}_{multi} = \arg\max_i \{\text{ROUGE-N}(r_i, s)\} \tag{6.5}$$

This measure is recall-oriented since it evaluates all the possible matches over the total number of n-grams occurring at the reference summary side.

ROUGE-L

This metric considers the Longest Common Sequence matches between every reference summary sentence, r_i and every candidate summary sentence c_j. The ROUGE-L metric is defined as follows:

$$ROUGE - L(S_1, S_2) = \frac{2 * R_{LCS-MEAD} * P_{LCS-MEAD}}{R_{LCS-MEAD} + P_{LCS-MEAD}} \tag{6.6}$$

where $R_{LCS-MEAD}$ is the maximum value of Long Common Sequence between the reference and the prediction obtained, over the length m of the reference summary:

$$R_{LCS-MEAD} = \frac{\sum_{s_{i1} \in S_1} \max_{s_{j2} \in S_2} LCS(s_i, s_j)}{m} \tag{6.7}$$

and $P_{LCS-MEAD}$ is the precision counterpart:

$$P_{LCS-MEAD} = \frac{\sum_{s_{i1} \in S_1} \max_{s_{j2} \in S_2} LCS(s_i, s_j)}{n} \tag{6.8}$$

where n is the length of the predicted summary.

This metric is popular in the summarization task, as a higher ROUGE score means a larger overlap of word units between references and predictions while a lower ROUGE score means a smaller overlap of word units.

There are some aspects that this metric does not into account. Firstly it treats each word equally, while some specific terms should be more relevant in the evaluation. Secondly, ROUGE does not assess how fluent the summary is. ROUGE only assesses the adequacy through n-grams, without considering possible synonyms and relationships among words.

Generally, extractive predictions will get higher ROUGE scores compared to abstractive predictions. The latter may use synonyms or in general words that may not be present in the references, hence producing a low ROUGE score despite being a good summary. It turns out that the choice of who created the reference summaries influences the usage of an extractive rather than an abstractive model.

According to [98], in the case of extractive summaries, this metric presents a low correlation with human judgments, slightly higher in cases for larger n, when managing speech transcripts. as the authors state, this low correlation, is probably caused by:

- Disfluencies in the dialogue summaries;
- The speaker information;
- Different stopword lists used in the metrics.

According to the authors, once these issues are fixed, or at least mitigated, even though the correlation between ROUGE and human judgment is still low, it can be considered acceptable.

BERTScore

To limit the underestimation of semantically correct pairs of candidate and reference texts, other researchers propose BERTScore [99], a task agnostic evaluation metric based on contextual embedding.

The score is computed as follows:

- First BERT model generates the contextual embeddings of both reference and candidate summaries, represented by a sequence of vectors $<\mathbf{x}_1, ..., \mathbf{x}_k>$ and $<\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_l>$;
- Then the similarity measure is computed among each token: in particular, the authors chose the pre-normalized cosine similarity $\mathbf{x}_i^T \hat{\mathbf{x}}_j$ to limit the computation to the inner product;

- Starting from the similarity measure, the score is computed as follows: the recall is given by the number of tokens in $\mathbf{x}$ that match with tokens in $\hat{\mathbf{x}}$ (more precisely, for each token in $\mathbf{x}$ the recall looks for the maximum match with all the tokens in $\hat{\mathbf{x}}$). Concerning the precision, the correspondence of each token in $\hat{\mathbf{x}}$ to a token in $\mathbf{x}$ is considered; the complete score is a f1-measure, the harmonic mean between precision and recall;

$$R_{BERT} = \frac{1}{|\mathbf{x}|} \sum_{x_i \in \mathbf{x}} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^T \hat{\mathbf{x}}_j \tag{6.9}$$

$$P_{BERT} = \frac{1}{|\hat{\mathbf{x}}|} \sum_{\hat{x}_i \in \hat{\mathbf{x}}} \max_{x_j \in \hat{x}} \mathbf{x}_i^T \hat{\mathbf{x}}_j \tag{6.10}$$

$$F_{BERT} = 2\frac{P_{BERT} \cdot R_{BERT}}{P_{BERT} + R_{BERT}} \tag{6.11}$$

- In order to incorporate the importance of each word with a weight, the authors adjust the recall and the precision measures with the corresponding IDF scores computed from the corpus;
- The score is re-scaled so that it lies between 0 and 1.

6.3 Results on punctuation

This section contains an ablation study on the results obtained for the punctuation restoration task. In particular, in section *Punctuation models on EduSum video lecture transcripts* the EduSum dataset has been employed to evaluate the goodness of the models on the MIT courses.

This section is followed by the paragraph *Qualitative evaluation on Politecnico video lecture transcripts* in which only a qualitative evaluation has been carried out on the Politecnico video lectures transcripts.

6.3.1 Punctuation models on EduSum video lecture transcripts

Motivated by the results on IWSLT2011 reported in the Transformer and Punctuator papers (displayed in Table 6.2), three attempts have been carried out in order to restore the punctuation of video lecture transcripts.

Results on IWSLT2011			
		Punctuator	Transformer
Comma	Precision	60.0	64.1
	Recall	45.1	68.8
	F1-score	51.5	66.3
Period	Precision	69.7	81.0
	Recall	69.2	83.7
	F1-score	69.4	82.3
Question mark	Precision	61.5	55.3
	Recall	45.7	74.3
	F1-score	52.5	63.4

Table 6.2: The table above summarizes the results on the test set reported in the Transformer and Punctuator papers.

FastPunct leads to reach better results with respect to the *Punctuator* only as highlighted in Figure 6.1, but remarkable improvements can be noticed with the Transformer architecture of Alam et Al. described in section *Transformer based models*.

The complete table of results is reported in Appendix A.1, Table A.1 and Table A.2. The barplot in 6.1 highlights a good property of transformer model: regardless of the nature of the course, the architecture is capable of returning good results overall. This robustness takes on a particular relevance in the case of Politecnico video lecture, which may contain formulae and specific terms unknown a priori by

the model.

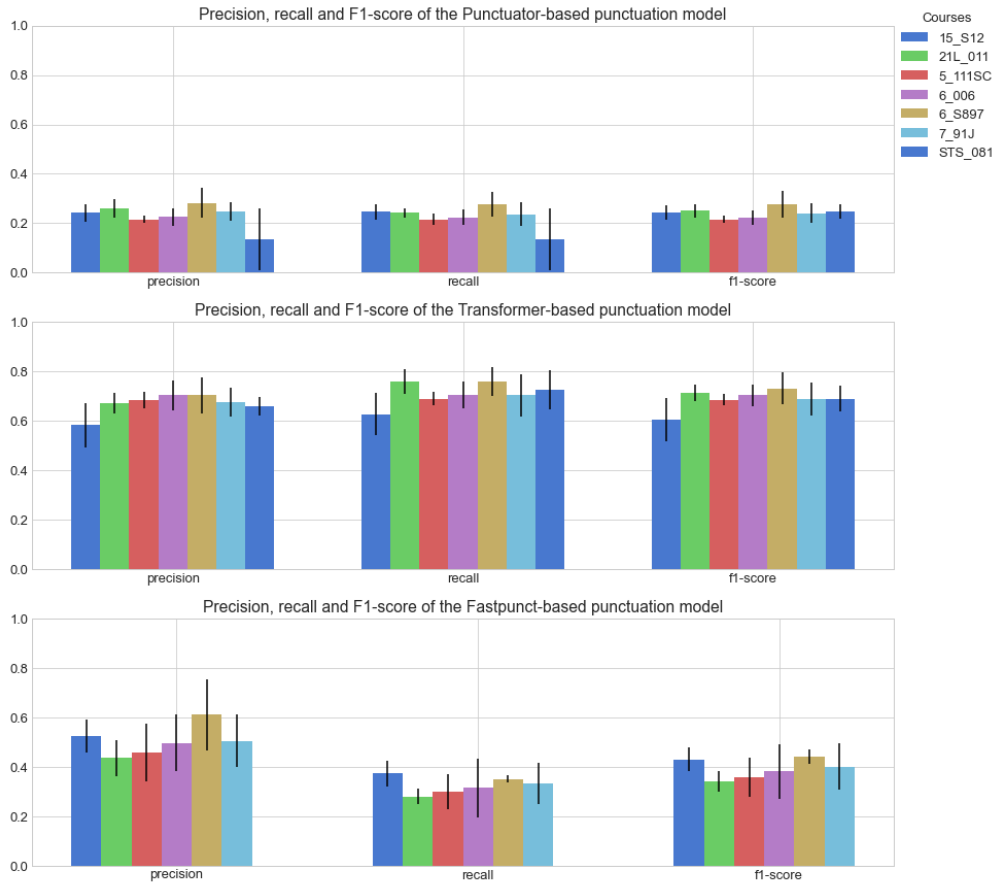

Figure 6.1: Summary of results (Precision, Recall and F1-score), with the different models over the courses.

Given these results, it is possible to consider the output of the transformer and proceed with the summarization task. An example of texts processed by the *Punctuator* architecture:

> *So Matt was putting a question on the table for us. So. why don't you go and lay that out And we'll get to the next [AUDIENCE]: This idea of how you distribute your talents across R&D sectors, The economic model*

that stands now is treat technology as a box and say, this Put, this much capital into technology and you'll get this kind of growth where it doesn't consider how you distribute your capital within that. [PROFESSOR]: I'll, just give you an example because you're, right in identifying the problem, About, seven or eight years ago, there were a whole series of major studies of climate. and how are we going to develop the necessary energy and related technologies that were going to be needed to tackle the problem? And? interestingly, all of those studies got to this moment where they were going to talk about the technologies And. they all basically read the same way. They all said, yes, we're going to need a lot of technologies. And then we'll spend a lot of money on them We'll spend a lot of money doing research and development was their conclusion

The same text, with *FastPunct* predictions:

So Matt was putting a question on the table for us, so why don't you go and lay that out and we'll get to the next [AUDIENCE]: this idea of how you distribute your talents across R&D sectors? The economic model that stands now is treat technology as [INAUDIBLE] a box and say this put this much capital into technology and you'll get this kind of growth. Where it doesn't consider how you distribute your capital within that [PROFESSOR]: I'll just give you an example because you're right in identifying the problem about seven or eight years ago. There were a whole series of major studies of climate and how are we going to develop the necessary energy and related technologies that were going to be needed to tackle the problem and interestingly. All of those studies got to this moment where they were going to talk about the technologies and they all basically read the same way they all said. Yes, we're going to need a lot of technologies, and then we'll spend a lot of money on them. We'll spend a lot of money doing research and development was their conclusion.

The corresponding output of RoBERTa-based model is:

So Matt was putting a question on the table for us. So why don't you go and lay that out, And we'll get to the next [AUDIENCE]: This idea of how you distribute your talents across R&D sectors, The economic model that stands now is treat technology as [INAUDIBLE] a box and say this, Put this much capital into technology, and you'll get this kind of growth where it doesn't consider how you distribute your capital within that. [PROFESSOR]: I'll just give you an example, because you're right in identifying the problem. About seven or eight years ago, there were a whole series of major studies of climate and how are we going to develop

the necessary energy and related technologies that were going to be needed to tackle the problem. And interestingly, all of those studies got to this moment where they were going to talk about the technologies, And they all basically read the same way. They all said, yes, we're going to need a lot of technologies, And then we'll spend a lot of money on them. We'll spend a lot of money doing research and development, was their conclusion.

6.3.2 Qualitative evaluation on Politecnico video lecture transcripts

The video lectures transcripts are relative to the Electronics Fundamentals and Applications course, and present some aspects that differ from the model's training dataset (IWSLT2011):

- specific terminology of the course;
- formulae and errors in the transcription of video lectures.

For a qualitative evaluation, see the chart in Figure 6.2: the first plot displays the percentage of commas, dots and question marks in the video lectures transcripts compared to those in the test set used in the papers [48, 24] (Figure 6.2, column *IWSLT2011 - ASR*). The second bar chart shows the mean number of words between two dots. From the former plot, it is possible to deduce that the *Punctuator* has a bias toward commas and the resulting sentences end up being longer than expected. This result is highlighted by the latter chart, where the bar associated with the *IWSLT2011 - ASR* is not even visible.

On average, the punctuated video lectures exhibit less than 5% of full stops, while more than 90% of symbols contained are commas. Higher results can are obtained with the *FastPunct* library.

As shown in Figure 6.3, the resulting percentage of commas and dots in the video lecture transcripts are in line with the ones in the test set, and the sentences generated are considerably shorter. From a readability standpoint, the Transformer punctuated texts are much more pleasant to read than the ones produced by the previous models. Some errors occur in the vicinity of formulae or variables: this is acceptable given the different nature of training set and video lecture texts. It is important to point out that the training dataset presents noticeable differences with respect to the video lectures, and the Transformer architecture turns out to be more robust than other models. Two extracts are reported below. Text punctuated by the *Punctuator* model:

okay, last part for the operational amplifier stages analysis is very simple is a very simple step from the previous analysis so is the design of active

filters active first order filters using the standard topologies that we already analyzed with the standard operational amplifier stages so essentially we wanted to design frequency response shapes using operation, amplifier and passive components connected to the operational amplifier as we already analyzed in the previous lessons last week, we can use also capacitors connected to the operational amplifier, obviously and starting from the differential stage and the different shade or stage in the integration stage we can easily move into the filter topology used to implement essentially one of these four classic theoretical shapes the first one is the the low pass filter shape this is the obviously the theoretical shape where we have the pass band region with a flat response

Same text processed by the Transformer architecture:

okay, last part for the operational amplifier, stages analysis is very simple. is a very simple step from the previous analysis. so is the design of active filters, active first order filters, using the standard topologies that we already analyzed with the standard operational amplifier stages. so essentially, we wanted to design frequency response shapes using operation amplifier and passive components connected to the operational amplifier, as we already analyzed in the previous lessons last week, we can use also capacitors connected to the operational amplifier, obviously. and starting from the differential stage and the different shade or stage, in the integration stage, we can easily move into the filter topology used to implement essentially one of these four classic theoretical shapes. the first one is the the low pass filter shape. this is the, obviously the theoretical shape, where we have the pass band region with a flat response.

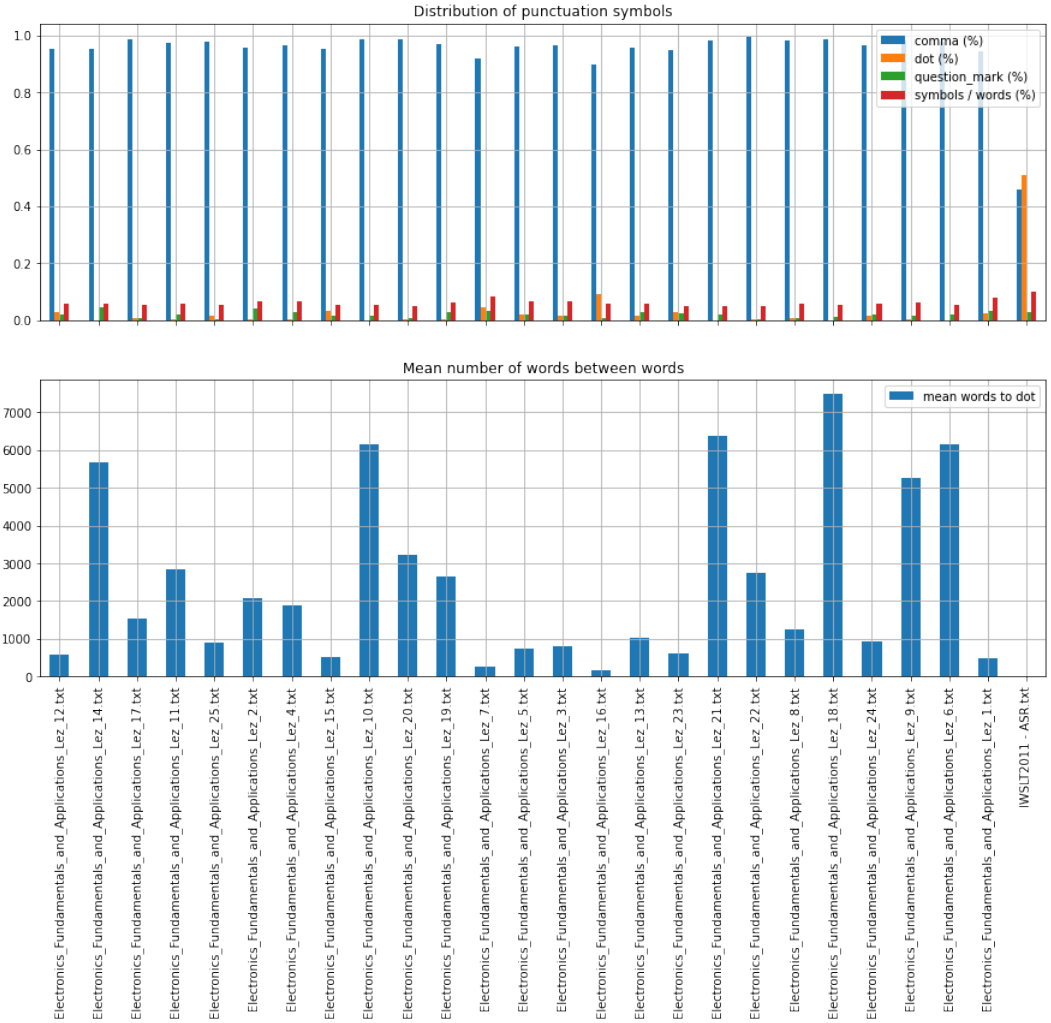

Figure 6.2: Distribution of punctuation marks obtained with the *Punctuator* (Bidirectional RNN presented in [48])

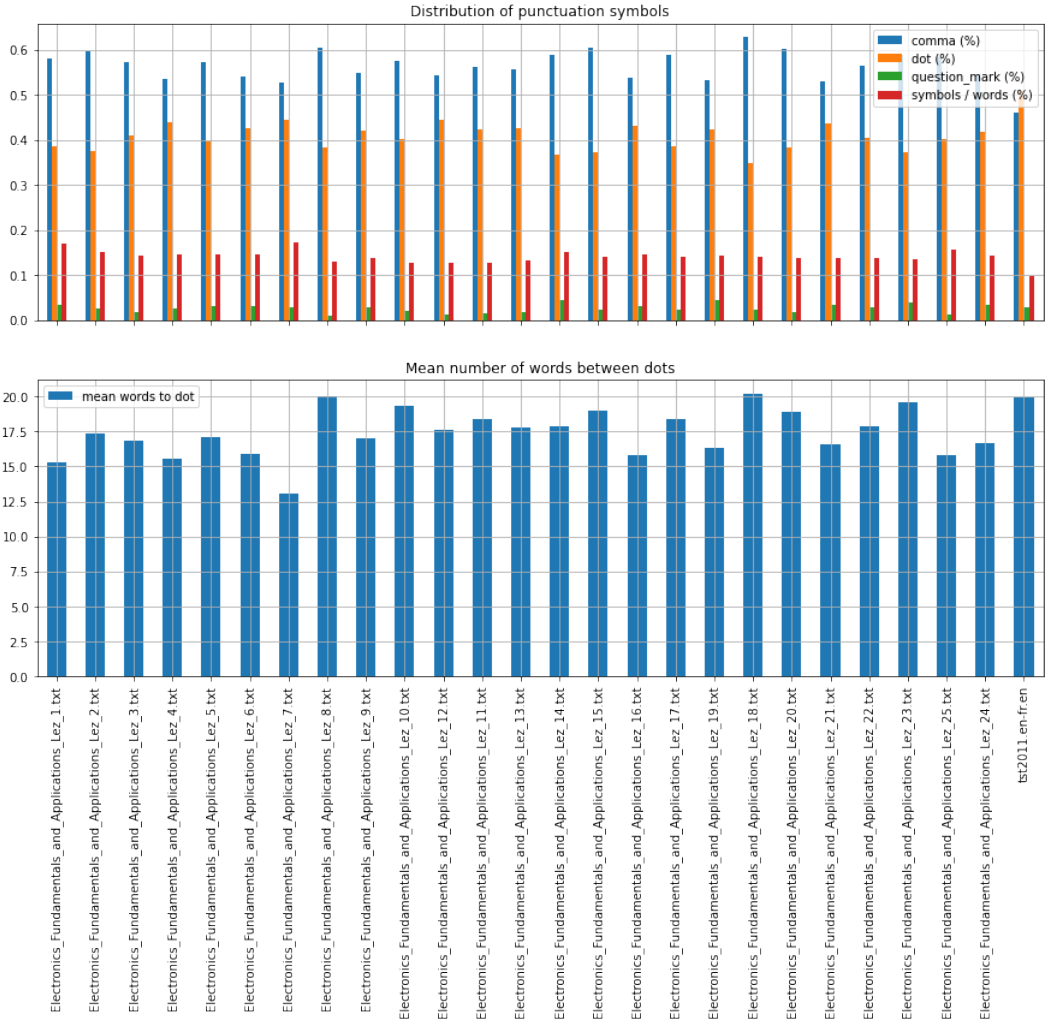

Figure 6.3: Distribution of punctuation marks obtained with the Transformer architecture (presented in [24])

6.4 Results on summarization

This section is devoted to analyzing the different components of each model in the two *results overview* sections.

The two architectures proposed are firstly evaluated in the meeting domain in the sections *Evaluation on the meeting dataset*, *Preliminary evaluation of extractive summaries on meetings* and *Evaluation of abstractive summaries on meetings*.

The first assumption questioned is the usefulness of the punctuation restoration task. To verify such assumption, the thesis dedicates two paragraphs *Evaluation on EduSum dataset: the impact of the punctuation* and *Evaluation of abstractive summaries on EduSumm: the impact of the punctuation* for HMNet model and the Extractive-Abstractive approach respectively. These sections evaluate the impact of punctuation symbols on the summaries generated by the different models.

The second assumption to verify is the benefit of the meeting-to-video lecture transfer learning: the analysis calls into question the generalization capability of fine-tuned model on meetings datasets with respect to the one only pretrained for general text summarization. This analysis is presented in *Evaluation on EduSum dataset: on the usage of the meeting set up and the impact of the subject* for HMNet architecture and in sections *Evaluation of abstractive summaries on EduSum: the impact of the subjects* and *On the usage of the meeting setting* for the proposed Extractive-Abstractive approach.

The last section *Comparison of the two models* makes a comparison between the two architecture presented.

6.4.1 HMNet: results overview

Evaluation on the meeting dataset

This paragraph evaluates HMNet with the modifications previously described. The results of the proposed modifications on HMNet are highlighted in Table 6.3. With the replacements of the speaker information with the topics, the drop in ROUGE-n metrics is consistent, but the results are still higher than the one discussed in the following section, with the Extractive-Abstractive model. Also, it is reasonable to consider that part of this drop may be also due to the reduced size of the output summaries. The hierarchical architecture is probably the strength of the model, since it can manage an entire meeting at a time without reduction of transcripts length.

Taking the same golden summary showed in Section *Extractive-Abstractive BART-based model* for the AMI dataset:

> *The Marketing Expert made a presentation on trend watching, including trends in user requirements and trends in fashion. The Industrial Designer*

Original version of HMNet			
	ROUGE-1	ROUGE-2	ROUGE-L
AMI	0.530 (0.495)	0.186 (0.170)	0.493 (0.489)
ICSI	0.463 (0.490)	0.106 (0.113)	0.474 (0.480)
Speaker-aware HMNet			
	ROUGE-1	ROUGE-2	ROUGE-L
AMI	0.462 (0.435)	0.155 (0.130)	0.428 (0.406)
ICSI	0.320 (0.317)	0.060 (0.064)	0.283 (0.285)
Results without finetuning			
	Test set (Validation set)		
	ROUGE-1	ROUGE-2	ROUGE-L
AMI	0.363 (0.345)	0.06 (0.07)	0.344 (0.350)
ICSI	0.289 (0.281)	0.044 (0.051)	0.277 (0.259)

Table 6.3: The table above summarizes the results on the test and development set (results of the development test within brackets) obtained on the AMI and the ICSI datasets with the original and modified version of HMNet (abstractive summaries). The so-called *speaker-aware* version uses the information of concepts rather than roles and limits the output summary to 200 tokens. The table reports also the results on the test and validation set (results of the development test within brackets) for the summarization task with HMNet pretrained on the CNN-Daily mail dataset, and on the AMI and the ICSI datasets. Comparing this table with the one highlighted in the HMNet paper it is possible to notice that the fine-tuning stage is necessary to improve meeting summaries.

> *presented all the components of the device and announced that several of the features already discussed would not be available. He suggested substituting a kinetic battery for the rechargeable batteries and using a combination of rubber and plastic for the materials. The User Interface Designer presented his main interface design, which included buttons for the most frequently used features and a graphic user interface on the LCD screen for other functions, to keep frequently used features easy to use. He announced that speech recognition was still an option to consider, depending on price. The Project Manager then began a discussion to decide what was going into the final design. It was decided that a kinetic battery would be used in place of a rechargeable battery, that the remote will feature an LCD screen and rubber casing and rubber buttons, and that interchangeable rubber covers in fruit colors will be available. Speech recognition may be included if it is not too costly.*

with the speaker-aware version of HMNet output this summary:

The marketing expert discussed the results of a user interface design study on consumer preferences and the availability of materials. The team then discussed the options for batteries and components and decided what materials to use in their product. The industrial designer discussed the interior workings of a remote and the team discussed various options for the casing and the components they would have to incorporate into the design. The group discussed not being able to use a kinetic battery because of their durability. The remote will be made of plastic. The case will be shaped like a voice recognition. The transient battery. Having a flip and volume on the channel will be active. The difference between keeping the remote simple, cool, ergonomic shape, and a fruit-inspired shape.

For the ICSI dataset, instead, given this golden summary:

The Berkeley Meeting Recorder group discussed efforts to train and test the Aurora group's HTK-based recognition system on ICSI's digits corpus. Members also discussed efforts to produce forced alignments from a selection of Meeting Recorder data. Performance in both tasks was adversely affected by the manner of recording conditions implemented and difficulties attributing utterances to the appropriate speakers. While debugging efforts resulted in improved forced alignments, dealing with mixed channel speech and speaker overlap remains a key objective for future work. The group is additionally focused on a continued ability to feed different features into the recognizer and then train the system accordingly.

The prediction of Microsoft researcher model is:

The berkeley meeting recorder group discussed the results of a set of digits data. The system was based on HKT, which the participants in the Aurora task were so similar they could be trained on. The results were finalized. However, the group leader would like to use the same data to elicit interaction and echo. There are worries regarding the ability of the group to test digits data across the set of the digits. The group also talked about the overlap diagnosis of the feature. A number of progress reports from the group for the transformations of the data collection were made in the past. This includes some discussion of the venerable data, including the authentication. The feature layer of the system, in turn, is a relatively small task and a small number of words.

The architecture fine-tuning is crucial for the performance on meeting dataset. An experiment is conducted in order to determine the benefits of the fine-tuning

stage on the architecture by generating meeting summaries with the pretrained model only. Results are displayed in Table 6.3, which shows that improvements of fine-tuning are consistent.

Evaluation on EduSum dataset: the impact of the punctuation

This paragraph is dedicated to understanding the impact of the punctuation restoration task on the summarization model, in this case HMNet architecture.

With EduSum dataset, the three setups compared are the following:

- The *original* configuration: the original text, with the correct punctuation are used as input text;
- The *automatic* setup: in which the transcription are filled with punctuation marks with the best model found in Section *Punctuation models on EduSum video lecture transcripts* and then summarized;
- The *no punctuation* configuration: in this case the input transcriptions completely lack punctuation.

The goal of this analysis is firstly to understand if the punctuation restoration is required as a preprocessing step for summarization; secondly, it also aims to verify the impact of such step.

Figure 6.4 and 6.5 show that again, from both ROUGE scores and BERTScore, it is possible to infer that the model is sufficiently robust even without punctuation. In this case, by inspecting the output summaries, it is possible to state that the lack of punctuation does not affect the summarization performance. In fact, the output summaries do not exhibit any newly introduced grammatical and syntactical mistakes. Consequently, it is likely that the errors introduced by the punctuation restoration task do not compromise the summarization task.

A complete report of the results is inserted in the appendix in A.3.

One issue that affects more or less all the summaries generated with HMNet model is that the last sentence is incomplete: the reason is that the model was trained to generate summaries of at least 400 characters for AMI and 420 for ICSI, while in this case this number is halved. Therefore, the model output reaches the maximum size without concluding the sentence it is currently producing.

An extract obtained with *Original* configuration:

> *The focus of the cornish show was on a 30-year decline in the of the continental era. The focus was on the dispossession of the western mythology industry. In the current issue, there are several forms of intolerance in the american over the past 25 years. The fact that there are historical and cultural explanations as well as an emotive issue. The bear disrepair*

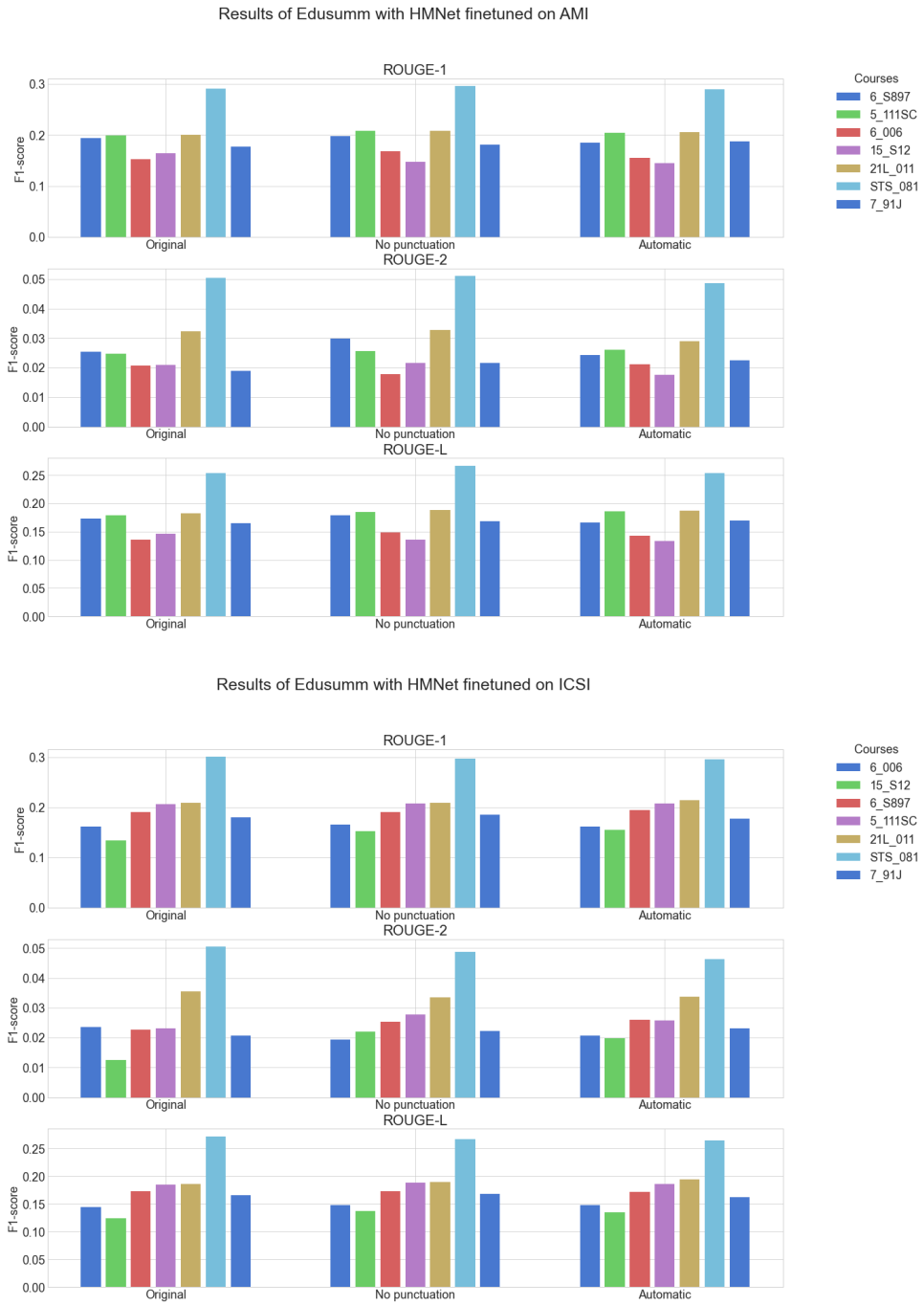

Figure 6.4: ROUGE scores obtained with Microsoft model finetuned on AMI (top) and ICSI (bottom) with EduSum dataset.

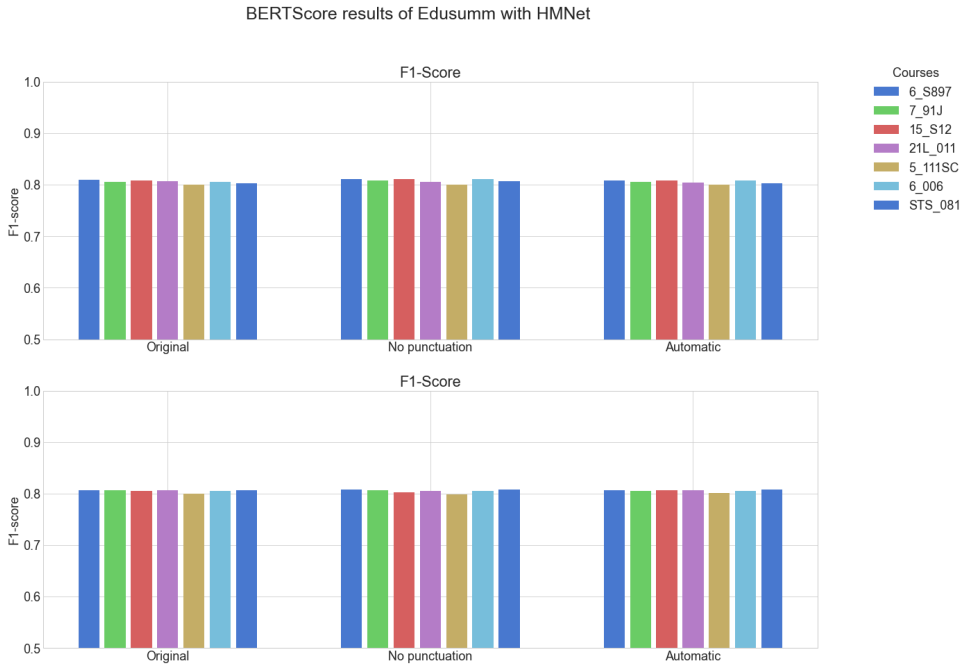

Figure 6.5: BERTScore results obtained with Microsoft model finetuned on AMI (top) and ICSI (bottom) with EduSum dataset.

> *in the u. Is an focus of what looks like the western values are seen as a problem for the westerns. It is unclear why the western genre is an issue , such as the western genre. The possibility of imposing a pauls as a native american cinema has been met with mixed reviews. There are historical events that will be carried out in the next meeting. The interpretations of the focus on the decaine of the hollywood system, and the corresponding of rhino violence, were also reflected in the use of subversive markers in the western american film. The film version of the superman in western parameters is open for genre, but it is not yet clear whether such parameters can be added to the notion that the movies and cinema are unsuitable in the future. In order to adapt into the and to the end, it was considered a seminal experiment that defines the 1863 for the western parameters of the*

The same input processed with the *No punctuation* setting:

> *The southerly of the western history complex was seen as a watershed moment in the western image industry. The fact that it is open spaces are seen as heretical, but the focus of the ongoing debate over western values and values of multiculturalism was on the mythological word. In*

the end, it was considered a focus on the dispossession of western image of the former mythology industry. There were several examples of such examples to be considered by western values like and : the cornish scene can see reflected in the and the dwells of the 1967 spell of focus. The holden article on the role of western movies in the current set of the joker , another adaptation of the classic western american cinema. The upcoming release will be presented to the group. After the end of the cold war , the group decided to adapt the notion of a western mythology show the western freedoms that such parameters are in decline. The prospect of descending on a scale like the huntsman article could be weighed against it. There was a corresponding correlation of view of the parameters that the action white supremacy in the time industry is expected to be from this stage. The possibility of incorporating the western genre into the western portrayal of the film industry was broader in terms of focus of research. The focus was on

Here is reported the corresponding output in the case of *Automatic* set up:

the upcoming film the wild west will be the focus of the western trend. The remote will be black. The film will be made of obsolete. The western genre is relatively simple : in western movies, in some sense, in relating to exorcism, materialism, and dispossession. Argues that western mythology may be used to convey the resentment of white countrymen, but also temperamental. He says the use of a focus on furthering the tyranny is an unlikely indicator; instead the western must be seen in some manner like a betrayal. He suggests that western, in particular, are muting itself and next door to reach the whole society. The project manager closes the meeting by telling each group member what his next task will be. The the industrial designer will work on the look and feel of the subject. The tribune will present the next play the western mythology. The western and western focus will be on the color. The biblical issue of western-style manners are subject to international media attention. The misrepresentation of a western mythology is a temptation to quantify, and enshrines the role of foreign policy as a mediator. Foreign policy may be restricting filming, but the possibility of using the word superhighway.

To conclude, the model is robust regardless of the presence of punctuation marks, and there no evidence of the importance of the punctuation restoration task in this case.

Evaluation on EduSum dataset: on the usage of the meeting set up and the impact of the subject

This section is devoted to evaluate the choice of fine-tuning on a meeting dataset and the difference in performance with respect to the course subjects.

It is expected that the meeting fine-tuning wuold be beneficial in the educational context, but unfortunately, in this case this assumption is not valid.

Firstly, it is important to point out that with HMNet the presence of sentences like "The project manager presents [...]" in the output summaries denotes the strong impact that the fine-tuning stage has on the entire architecture. In fact, it is possible to verify that there aren't any marketing experts or project managers involved in EduSum dataset, and this behavior is attributable to the training procedure on the AMI and the ICSI, where their presence is known. As far as the subjects are concerned, the course that benefits from the AMI and the ICSI finetuning the most is the economics/innovation course (STS-081). In this case, the model performances are consistently higher than the one obtained on other courses.

The cause of that may be the finetuning stage: the AMI and the ICSI datasets report the summarization of meeting in economics jargon. Probably, the vocabulary present in the STS-081 course shares some terms with the meetings datasets. The model, being strongly influenced by this fine-tuning, is capable of generating better results. Figure 6.6 illustrates how the results of HMNet in the case of STS-081 increase when the model is finetuned with the two meeting dataset.

To conclude, the benefits from this architecture are counterbalanced by the increased size in the output summaries, and the implementation of BART with extractive can still be considered a valuable option for this task. Noticeable improvements with STS-081 can be seen in all of the metrics considered, in both the AMI and the ICSI finetuning cases.

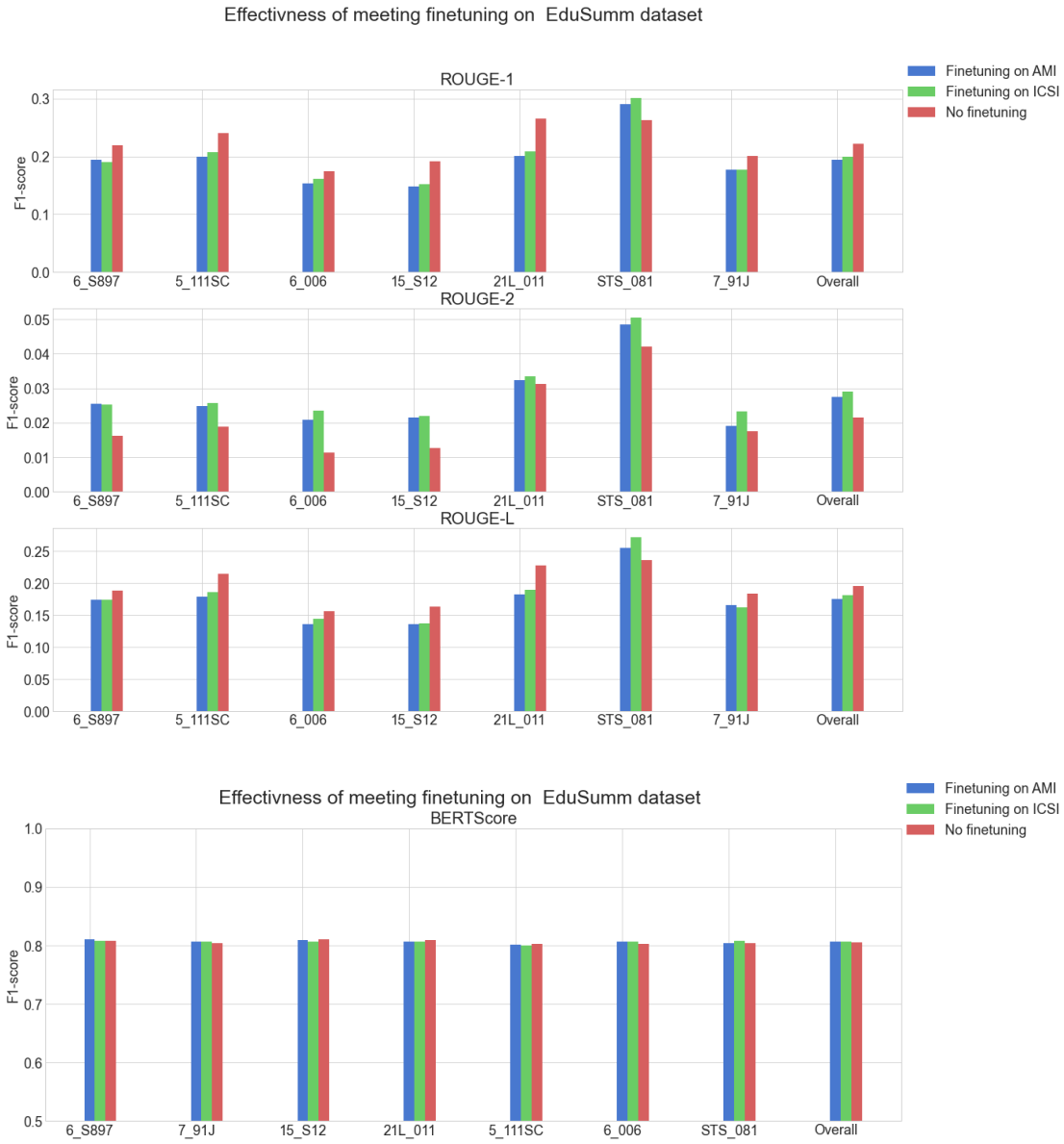

Figure 6.6: Results comparison obtained with EduSum with HMNet finetuned on AMI, on ICSI and not finetuned

6.4.2 Extractive-Abstractive: results overview

Preliminary evaluation of extractive summaries on meetings

In order to estimate the goodness of the extractive algorithms chosen before BART architecture, the resulting extractive summaries are evaluated and compared with the ones provided by the AMI dataset. Unfortunately, similar evaluations on ICSI dataset were not possible, as in this case the gold summaries were not available.

Results are displayed in Table 6.4: LSA tends to favor ROUGE-1 score, while Truncation and TextRank exceed the former in the case of larger n. A possible explanation of this phenomenon can be derived from the following claim by the authors of [65]: "graph-based ranking algorithm is a way of deciding on the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph". The selected sentences are the most recommended by all of the sentences in the entire meeting, meaning that they are the most similar in terms of cosine similarity. TextRank sentences turn out to be shorter: since the overall text has a fixed total length, the sentence contained are more numerous.

In fact, it is possible to notice that the sentences extracted with TextRank contain on average 10 words, while LSA selects sentences of 28 words on average. The more sentences are included, the more it is likely to get a sentence included in the summary.

Extractive summarization results on AMI dataset			
	Test set (Validation set)		
	ROUGE-1	ROUGE-2	ROUGE-L
LSA	0.706 (0.738)	0.497 (0.563)	0.237 (0.259)
TextRank	0.663 (0.683)	0.440 (0.486)	0.380 (0.433)
BERT	0.67 (0.6965)	0.4507 (0.4953)	0.2261 (0.2441)
Random	0.635 (0.660)	0.376 (0.407)	0.221 (0.229)
Truncation	0.674 (0.674)	0.471 (0.471)	0.327 (0.327)
Mix	0.527 (0.570)	0.5027 (0.547)	0.5165 (0.563)

Table 6.4: The table shows the results on the test set and on the validation set (between brackets) for the extractive summarization task with different techniques placed upstream BART. In this case, the ground truth compared is the extractive, present in the AMI dataset.

Evaluation of abstractive summaries on meetings

The goal of this section is to measure the capability of Extractive-Abstractive architecture in adapting its structure for meetings, where the literature provides

the two well-known dataset previously described.

The assumption is that, if the entire structure is capable of generating good results in the case of meetings, it would probably be able to achieve interesting results for video lecture transcripts as well, even without further training (also considering the limited number of examples of video lecture summaries).

Results on the meeting dataset with the approach previously described are shown in Table 6.5: the improvement given by the finetuning reaches a +0.07. Thus, from this table it is possible to state that Extractive-Abstractive model outputs similar results regardless of the upstream extractive summarization techniques: comparing the results obtained with the extractive methods with respect to the *Precomputed* ones (the extractive summaries provided by the dataset - only available for the AMI dataset), it is possible to notice that the results are similar, slightly better in the case of LSA or TextRank. Moreover, further improvements can be obtained improving on the abstractive model rather than exploiting better extractive summarization techniques. In the ICSI results, the entry *Precomputed* is missing as the extractive summaries were not included in the dataset. For both datasets, LSA and TextRank methods seem to be the most effective preprocessing step, allowing the model to reach higher performances.

An example of summary, coming from AMI dataset is the following obtained with TextRank preprocessing:

> *The operating system of the remote control was designed to be easy to use, but the controls were made of rubber buttons that they decided to use in the form of a screen or a touch display. The key part of the design was to limit the size of the devices and to increase the amount of space needed for the necessary functions. The major issue with the ALCA function was to eliminate the need for the large number of the power-related functions in the remote control.*

Instead, the example below, is generated with LSA preprocessing:

> *The remote controls are designed to be in a standby mode. They need to be made available in the original and similar design, but they would have to be kept separate from the rest of the products in the market. The key part was to use the the following material for the purpose of keeping the the control functions consistent with the AEC central pattern. They were concerned about the the need to use voice recognition at the base of the device.*

The corresponding gold summary is:

> *The Marketing Expert made a presentation on trend watching, including trends in user requirements and trends in fashion. The Industrial Designer*

Best results on AMI dataset			
	Test set (Validation set)		
	ROUGE-1	ROUGE-2	ROUGE-L
LSA	0.362 (0.374)	0.075 (0.070)	0.221 (0.214)
TextRank	0.364 (0.378)	0.075 (0.0855)	0.210 (0.221)
BERT	0.3676 (0.36)	0.074 (0.075)	0.20 (0.21)
Random	0.337 (0.322)	0.053 (0.067)	0.185 (0.177)
Truncation	0.362 (0.373)	0.076 (0.082)	0.222 (0.210)
Mix	0.317 (0.335)	0.066 (0.070)	0.195 (0.206)
Precomputed	0.361 (0.358)	0.087 (0.086)	0.213 (0.201)
Best results on ICSI dataset			
	Test set (Validation set)		
	ROUGE-1	ROUGE-2	ROUGE-L
LSA	0.251 (0.245)	0.041 (0.059)	0.164 (0.156)
TextRank	0.263 (0.268)	0.032 (0.04)	0.145 (0.157)
BERT	0.2 (0.24)	0.04 (0.03)	0.134 (0.14)
Random	0.259 (0.250)	0.025 (0.030)	0.130 (0.138)
Truncation	0.240 (0.248)	0.016 (0.029)	0.123 (0.131)
Mix	0.255 (0.264)	0.034 (0.038)	0.015 (0.162)
Results on AMI without finetuning			
	Test set (Validation set)		
	ROUGE-1	ROUGE-2	ROUGE-L
LSA	0.290 (0.311)	0.061 (0.06)	0.154 (0.174)
TextRank	0.291 (0.316)	0.052 (0.063)	0.158 (0.160)
BERT	0.29 (0.30)	0.054 (0.061)	0.15 (0.167)
Results on ICSI without finetuning			
LSA	0.251 (0.248)	0.026 (0.032)	0.141 (0.140)
TextRank	0.261 (0.250)	0.013 (0.033)	0.142 (0.141)
BERT	0.221 (0.211)	0.030 (0.028)	0.131 (0.123)

Table 6.5: The table summarizes the results on the test and validation set (between brackets) for the summarization task with different extractive techniques placed upstream BART on AMI and ICSI dataset. In the first case, LSA and TextRank are capable of reaching the results obtained when the extractive summary is the best possible (the *Precomputed* case). With ICSI scores are lower than the AMI dataset, and the differences among the pre-processing phase are less evident. Also the table reports the results on the test and validation set (between brackets) for the meeting summarization task with different extractive techniques placed upstream BART on AMI and ICSI dataset with BART model not finetuned on any dataset. Comparing these results it is possible to notice that finetuning is necessary for improving meeting summaries.

presented all the components of the device and announced that several of the features already discussed would not be available. He suggested substituting a kinetic battery for the rechargeable batteries and using a combination of rubber and plastic for the materials. The User Interface Designer presented his main interface design, which included buttons for the most frequently used features and a graphic user interface on the LCD screen for other functions, to keep frequently used features easy to use. He announced that speech recognition was still an option to consider, depending on price. The Project Manager then began a discussion to decide what was going into the final design. It was decided that a kinetic battery would be used in place of a rechargeable battery, that the remote will feature an LCD screen and rubber casing and rubber buttons, and that interchangeable rubber covers in fruit colors will be available. Speech recognition may be included if it is not too costly.

Another example coming from the ICSI dataset, where the gold summary is represented by:

The Berkeley Meeting Recorder group discussed efforts to train and test the Aurora group's HTK-based recognition system on ICSI's digits corpus. Members also discussed efforts to produce forced alignments from a selection of Meeting Recorder data. Performance in both tasks was adversely affected by the manner of recording conditions implemented and difficulties attributing utterances to the appropriate speakers. While debugging efforts resulted in improved forced alignments, dealing with mixed channel speech and speaker overlap remains a key objective for future work. The group is additionally focused on a continued ability to feed different features into the recognizer and then train the system accordingly.

And the BART summary with TextRank prepocessing step:

There was a messy alignment process where the group had to make up a separate set of numbers for the test set. However, they were unable to perform the task because there was a large amount of data and training in the SRI system. They needed to equip the trainee with a pair of arm - mounted telephones or a head-mounted cell phone or a combination of them. The challenge was to try out the program at the front-end meeting on Saturday night.They agreed to do the task based on HTK, which was based on HTK, that was used by all the participants in Aurora.

While the BART output preceded by the LSA extraction:

The group agreed that they need to be able to improve on the forced alignments based on HTK extensions in order to increase the performance.The group decided to test out the demands of the crowd by using a telephone bandwidth front-end extension to the base-end extension to address the needs of the group. They agreed to try to gain more control over the forced aligned alignments compared to the rest of the groups at present. They agreed to help resolve the issues related to the obstruction issues associated with the forced alignments when they were brought up to limit the number of interruptions in the area where the obstructions were present.

Note that the summary generated is much shorter than the gold summary: the minimum and the maximum length of the output summary are set to 100 and 200 respectively, as in the educational dataset the expected results range between these two bounds.

Evaluation of abstractive summaries on EduSumm: the impact of the punctuation

This section presents an analysis conducted to determine the impact of the punctuation restoration task on the summary task with EduSumm dataset. To do so only the courses whose transcriptions with the correct punctuation were available are employed.

The goal of this analysis is to verify if a punctuation restoration is worthwhile for the summarization purpose and if this step is significant for the summarization model performances.

Authors in [100] found that, for a multi-class classification task, transformer-based language models are robust to changes in punctuation. The analysis conducted will demonstrate that, in this case, such statement could be questionable.

For this analysis, three configurations are compared:

- The results obtained with the original transcriptions, with the correct punctuation, later called *original*.
- The outputs obtained with the transcriptions, whose punctuation is automatically restored; these transcriptions only contains full stops, question marks and commas. In this case it is possible to measure the gravity of errors introduced with the first phase. This configuration will be under the name of *automatic*.
- The results obtained with the transcription whose punctuation marks have been completely removed: in this case it is possible to understand if a language models is robust enough and correctly introduces punctuation in the summary generated even though the input texts are unpunctuated. This will be referred as *no punctuation*.

The models used are the best ones found in the previous section: the Extractive-Abstractive which employs LSA and TextRank algorithms as extractive algorithm. Results in terms of ROUGE-score are reported in Figure 6.7 and 6.8. The BERTScore results are displayed in Figure 6.9.

At first glance, excluding some cases, it would seem that the punctuation restoration task is of little value.

However, when the processed texts do not contain punctuation marks, the resulted summaries are more prolix than the one obtained with the original text, and present some errors that affect the punctuation. In these cases, the output sentence is too long and therefore is cut as soon as the model reaches the maximum length, and the final outcome does not contain any punctuation marks.

In fact, on average, the length of summaries that are generated from unpunctuated text is about 120 ± 33 words, while do not exceed 104 ± 20 words for the summaries coming from the original text or the ones that are generated from text punctuated automatically. A complete report of the results is inserted in Appendix A.2.

An example of output is reported below, obtained with the correct punctuation:

> *The Western mythologies operating in the United States today functioned as central functions of the Western culture. Therefore, they need to reframe the notion that the Western myths originated in the Western cultures and were associated with the Central Powers of the Culture.They need to resolve the internal divisions in the cultural organization over the issue of guns and gun control. They need to settle the internal disputes over the issues related to the Western Mythologies based on the root causes of the political unrest in the West.*

This is the counterpart obtained from the non-punctuated text in input:

> *The Western originated at the end of a period in which the Western was established as a branch of the Western Empire. In fact, the Western form was born at the beginning of a series of events in the Western History titled "The Significance of the Frontier in American History". In the present context the Western forms are associated with the Western based on the claims of the wild West respectively. In the present course the Western form is related to the demands of the Wild West Form intertwined with the claim of the Northern Territory and under the influence of the Central Powers of the U.S.A.widely split into separate groups due to the confining the lives of women in the western mass differently separated from the rest of the society through the extension of the Christian Culture attached to the Western-based organization functioned in the molding of the*

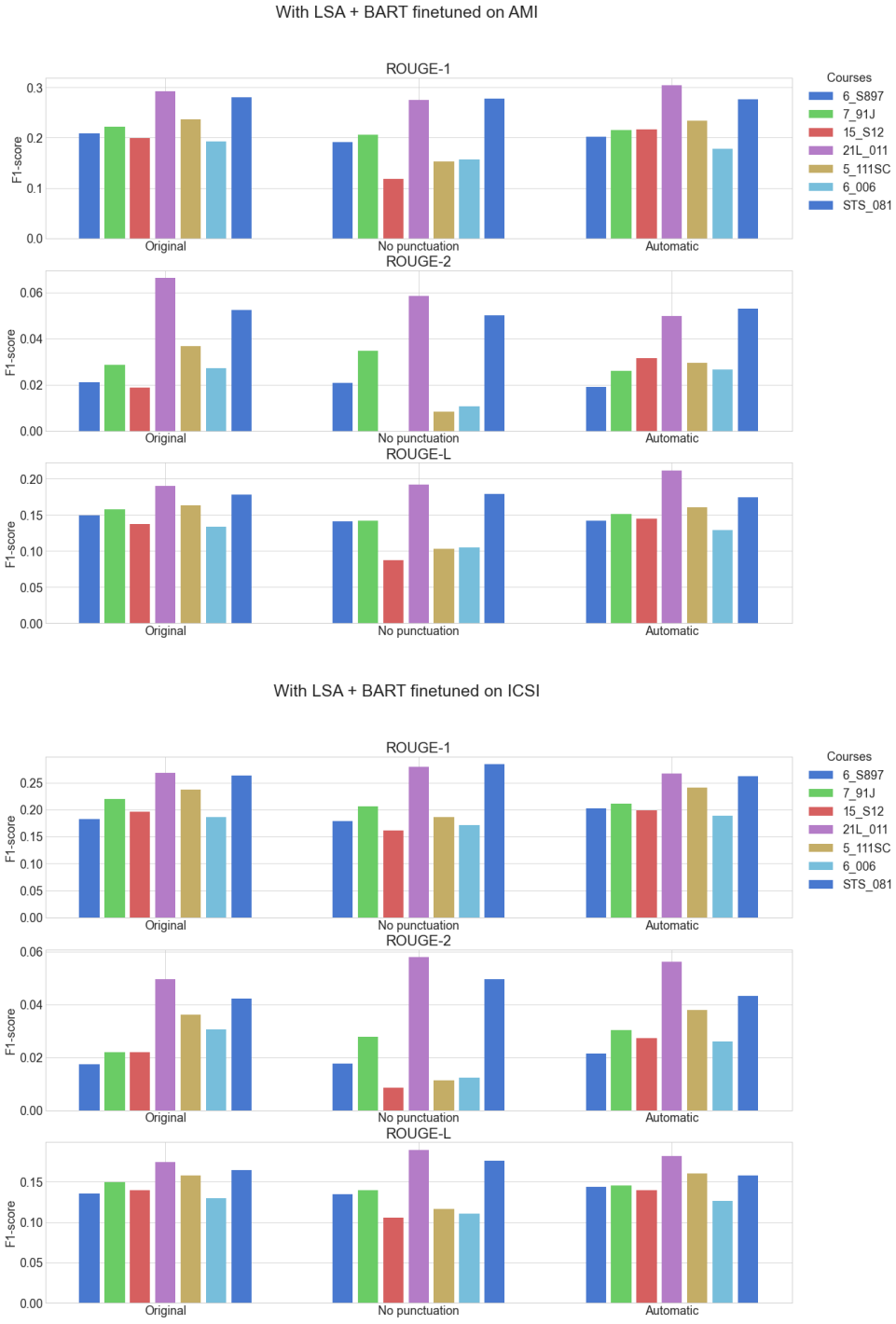

Figure 6.7: Results obtained on EduSum dataset, with LSA before BART model finetuned on the AMI (top) and the ICSI (bottom) datasets.

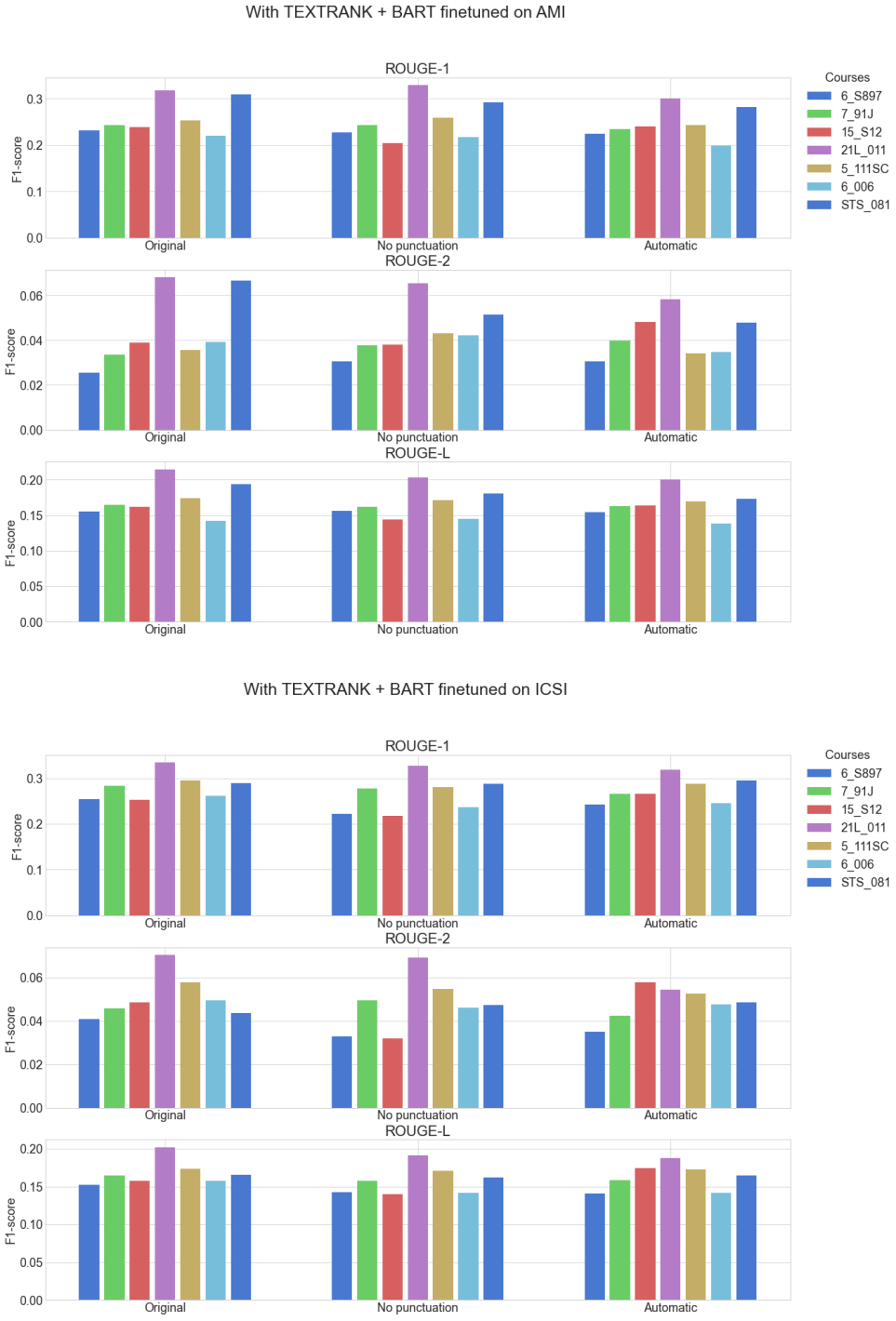

Figure 6.8: Results obtained on EduSum dataset, with TextRank before BART model finetuned on the AMI (top) and the ICSI (bottom) datasets.

Figure 6.9: BERTScore Results obtained on EduSum dataset with LSA or TextRank before BART model finetuned on the AMI and the ICSI datasets.

While the summary generated from the text punctuated by the Transformer is:

> *The American Civil War fought against the United States over the issue of guns and gun control in the Western milieu. There was a division between the cultures associated with the Western myths and the traditions associated with them at the time of the advent of the Western films.In fact, there were separate groups related to either the Western or the Christian customs associated with 'The Western' based on the ancient Greek Mythologies associated with Western culture.*

The reason of this degradation is twofold. The lack of punctuation firstly affects the extractive component of the pipeline: the full stops determine the sentence boundaries with which the extractive algorithm breaks the large corpus, and for each sentence, the extractive algorithm computes a score. With the removal of the punctuation and division of the text in sequence of $N = 30$ tokens, there is a small drop in performance in the extractive summaries generation process. The following results are the outcome of an experiment conducted with AMI dataset, whose extractive summaries were provided. Figure 6.10 compares the results obtained with the two best extractive algorithms (LSA and Textrank) in the case when the text is provided with the correct punctuation (*Original* configuration), when the text lacks punctuation (*No punctuation* configuration) and when the punctuation is restored with the transformer architecture (*Transformer* setup).

From these plots, it is possible to notice that the quality of the extractive summaries degrades in the proximity of texts without punctuation. Such effects probably propagate into the abstractive model. For the reasons explained in Section *Summary evaluation metrics*, these errors cannot be captured with the ROUGE score only, but require more careful examination.

In short, the punctuation restoration task is necessary, first of all to improve the sentence extraction upstream BART model, and secondly to obtain more readable outputs with the abstractive model.

Evaluation of abstractive summaries on EduSum: the impact of the subjects

The courses analyzed in the Edusum dataset are of different nature, from humanistic to scientific courses, with different vocabularies and characteristics. This paragraph is devoted to evaluating the effects of the different class subjects in the generated summaries.

According to Table 6.6, 6.7 and 6.8, which show the results according to the class subject, the results obtained on Chemistry (5-111SC) course are not the worste, despite such course being rich in formulae. Interestingly enough, the model is capable of generating summaries that contain few of them. And inspecting the

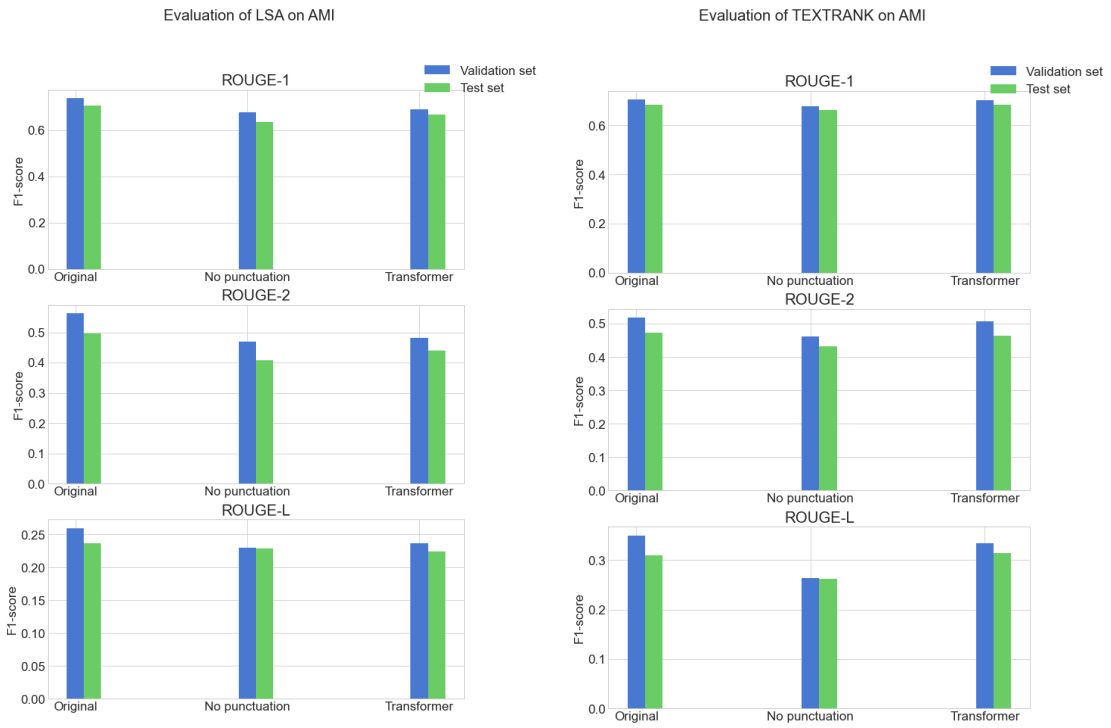

Figure 6.10: Extractive summaries results obtained with LSA (left) and TextRank (right) with different punctuation setups.

extractive summaries that feed the abstractive model, it is possible to see that most formulae are already filtered out at this stage. Most likely, for scientific subjects, the extractive summarization applies a good filter for removing sentences with formulae. In general, scientific subjects are more difficult to summarize, since they have a more technical language that the models ignore during training. In fact, the lowest results are obtained with 6-006 course, which deals with Algorithm and Computer science, and 15-S12, the course focused on Blockchains.

In Section *Punctuation models on EduSum video lecture transcripts*, for punctuation restoration task, a little drop in F1-score with the transformer model on course STS-081 is seen. Table 6.6, 6.7 and 6.8 show that, on the contrary, this course seems to be one of the easiest to summarize. Perhaps, all other courses with the exception of STS-081 have a specific vocabulary different from the AMI and the ICSI dataset, while the economics jargon of the Innovation course has some overlaps with the two and therefore in this case the results are higher.

Course	Metrics		Models			
			TextRank-BART		LSA-BART	
			AMI	ICSI	AMI	ICSI
21L-01	ROUGE-1	Precision	0.343	0.387	0.339	0.341
		Recall	0.292	0.281	0.268	0.221
		F1-score	**0.313**	**0.322**	**0.298**	**0.266**
	ROUGE-2	Precision	0.083	0.087	0.061	0.063
		Recall	0.072	0.064	0.05	0.04
		F1-score	**0.076**	**0.073**	**0.054**	**0.049**
	ROUGE-L	Precision	0.231	0.227	0.226	0.2368
		Recall	0.197	0.165	0.178	0.151
		F1-score	**0.211**	**0.189**	**0.198**	**0.183**
	BERTScore	Precision	0.846	0.842	0.836	0.83
		Recall	0.837	0.842	0.833	0.833
		F1-score	**0.841**	**0.842**	**0.834**	**0.831**
STS-081	ROUGE-1	Precision	0.261	0.264	0.253	0.237
		Recall	0.39	0.369	0.368	0.348
		F1-score	**0.303**	**0.292**	**0.288**	**0.269**
	ROUGE-2	Precision	0.05	0.043	0.051	0.037
		Recall	0.073	0.06	0.075	0.058
		F1-score	**0.057**	**0.048**	**0.058**	**0.043**
	ROUGE-L	Precision	0.153	0.151	0.153	0.147
		Recall	0.229	0.215	0.227	0.216
		F1-score	**0.177**	**0.168**	**0.175**	**0.166**
	BERTScore	Precision	0.846	0.849	0.839	0.836
		Recall	0.825	0.831	0.82	0.822
		F1-score	**0.835**	**0.84**	**0.829**	**0.829**
5-111SC	ROUGE-1	Precision	0.311	0.374	0.292	0.34
		Recall	0.217	0.249	0.196	0.196
		F1-score	**0.252**	**0.293**	**0.231**	**0.244**
	ROUGE-2	Precision	0.055	0.074	0.039	0.062
		Recall	0.038	0.047	0.027	0.033
		F1-score	**0.044**	**0.056**	**0.031**	**0.042**
	ROUGE-L	Precision	0.213	0.216	0.204	0.223
		Recall	0.147	0.144	0.136	0.126
		F1-score	**0.171**	**0.169**	**0.16**	**0.158**
	BERTScore	Precision	0.834	0.833	0.816	0.826
		Recall	0.837	0.844	0.833	0.834
		F1-score	**0.835**	**0.839**	**0.824**	**0.83**

Table 6.6: Results of summarization task obtained with different version of the proposed model on the courses of the EduSum dataset, with models pretrained on AMI and ICSI.

Course	Metrics		Models			
			TextRank-BART		LSA-BART	
			AMI	ICSI	AMI	ICSI
7-91J	ROUGE-1	Precision	0.308	0.356	0.302	0.305
		Recall	0.203	0.24	0.18	0.177
		F1-score	**0.242**	**0.283**	**0.222**	**0.22**
	ROUGE-2	Precision	0.043	0.057	0.039	0.03
		Recall	0.028	0.039	0.023	0.018
		F1-score	**0.034**	**0.046**	**0.029**	**0.022**
	ROUGE-L	Precision	0.21	0.207	0.216	0.207
		Recall	0.137	0.14	0.127	0.121
		F1-score	**0.164**	**0.165**	**0.157**	**0.15**
	BERTScore	Precision	0.832	0.839	0.824	0.824
		Recall	0.839	0.848	0.833	0.834
		F1-score	**0.835**	**0.843**	**0.829**	**0.829**
15-S12	ROUGE-1	Precision	0.345	0.361	0.3	0.29
		Recall	0.188	0.201	0.154	0.154
		F1-score	**0.238**	**0.253**	**0.2**	**0.197**
	ROUGE-2	Precision	0.059	0.072	0.031	0.034
		Recall	0.03	0.038	0.014	0.017
		F1-score	**0.039**	**0.049**	**0.019**	**0.022**
	ROUGE-L	Precision	0.236	0.226	0.208	0.211
		Recall	0.127	0.125	0.105	0.108
		F1-score	**0.162**	**0.157**	**0.137**	**0.14**
	BERTScore	Precision	0.832	0.839	0.824	0.824
		Recall	0.849	0.857	0.842	0.843
		F1-score	**0.845**	**0.85**	**0.839**	**0.836**
6-006	ROUGE-1	Precision	0.325	0.396	0.302	0.321
		Recall	0.168	0.198	0.143	0.133
		F1-score	**0.249**	**0.261**	**0.232**	**0.227**
	ROUGE-2	Precision	0.059	0.077	0.043	0.055
		Recall	0.03	0.037	0.02	0.021
		F1-score	**0.039**	**0.05**	**0.027**	**0.031**
	ROUGE-L	Precision	0.211	0.24	0.21	0.224
		Recall	0.109	0.119	0.099	0.093
		F1-score	**0.142**	**0.158**	**0.133**	**0.13**
	BERTScore	Precision	0.83	0.832	0.827	0.823
		Recall	0.843	0.853	0.842	0.843
		F1-score	**0.836**	**0.843**	**0.834**	**0.833**

Table 6.7: Results of summarization task obtained with different version of the proposed model on the courses of the EduSum dataset, with models pretrained on AMI and ICSI.

Course	Metrics		Models			
			TextRank-BART		LSA-BART	
			AMI	ICSI	AMI	ICSI
6-S897	ROUGE-1	Precision	0.325	0.36	0.288	0.286
		Recall	0.183	0.202	0.168	0.138
		F1-score	**0.232**	**0.254**	**0.209**	**0.183**
	ROUGE-2	Precision	0.036	0.054	0.028	0.026
		Recall	0.02	0.034	0.017	0.013
		F1-score	**0.026**	**0.041**	**0.021**	**0.017**
	ROUGE-L	Precision	0.22	0.218	0.206	0.216
		Recall	0.122	0.121	0.12	0.102
		F1-score	**0.155**	**0.152**	**0.15**	**0.136**
	BERTScore	Precision	0.839	0.833	0.834	0.823
		Recall	0.839	0.843	0.836	0.835
		F1-score	**0.839**	**0.837**	**0.835**	**0.829**

Table 6.8: Results of summarization task obtained with different version of the proposed model on the courses of the EduSum dataset, with models pretrained on AMI and ICSI.

On the usage of the meeting setting

This paragraph presents an examination of the importance of fine-tuning on the AMI and the ICSI datasets. This analysis is conducted comparing between the summaries generated with the Extractive-Abstractive approach fine-tuned on the AMI/ICSI and the same model pretrained only on SAMSum dataset.

As previously stated, for the AMI dataset in particular, the fine-tuning allows obtaining improvements of +0.07, + 0.02 and +0.06 on ROUGE-1, ROUGE-2 and ROUGE-L respectively. It would be reasonable to observe similar improvements on educational data. The bar plots in Figure 6.11 and 6.12 show minor improvements, mostly regarding course STS-081 with AMI fine-tuning. For all other cases, the fine-tuning seems to not improve the results. This result confirms what the authors in [81] found: if the model pretraining is robust enough, the improvements with further training are limited to a certain range of domains. Besides, the AMI and ICSI datasets are relatively small and their impact on the training can be observed only in the meeting domain.

A notable example is reported below: a summary for 21L-011 course, generated without fine-tuning the model contains the concept of “consensus narrative”. No other models were capable of doing that.

The Hollywood system is committed to genre reforms. The movies and

the Western form embody this notion of a story form that tends to reach the whole society or to speak for the whole people. The format of the movies are often conflicted and divided, but it's important to understand the cultural work of the movie as a part of the larger process of the history of the cinema. It's also important to recognize the importance of genre reforms and the power of the culture in which the movies play. For example, the use of the term "consensus narrative" suggests that the movies become a form of central storytelling in the society and the culture engages in an ongoing conversation about issues such as guns and gun control.

The output summary generated with LSA as extractive before BART.

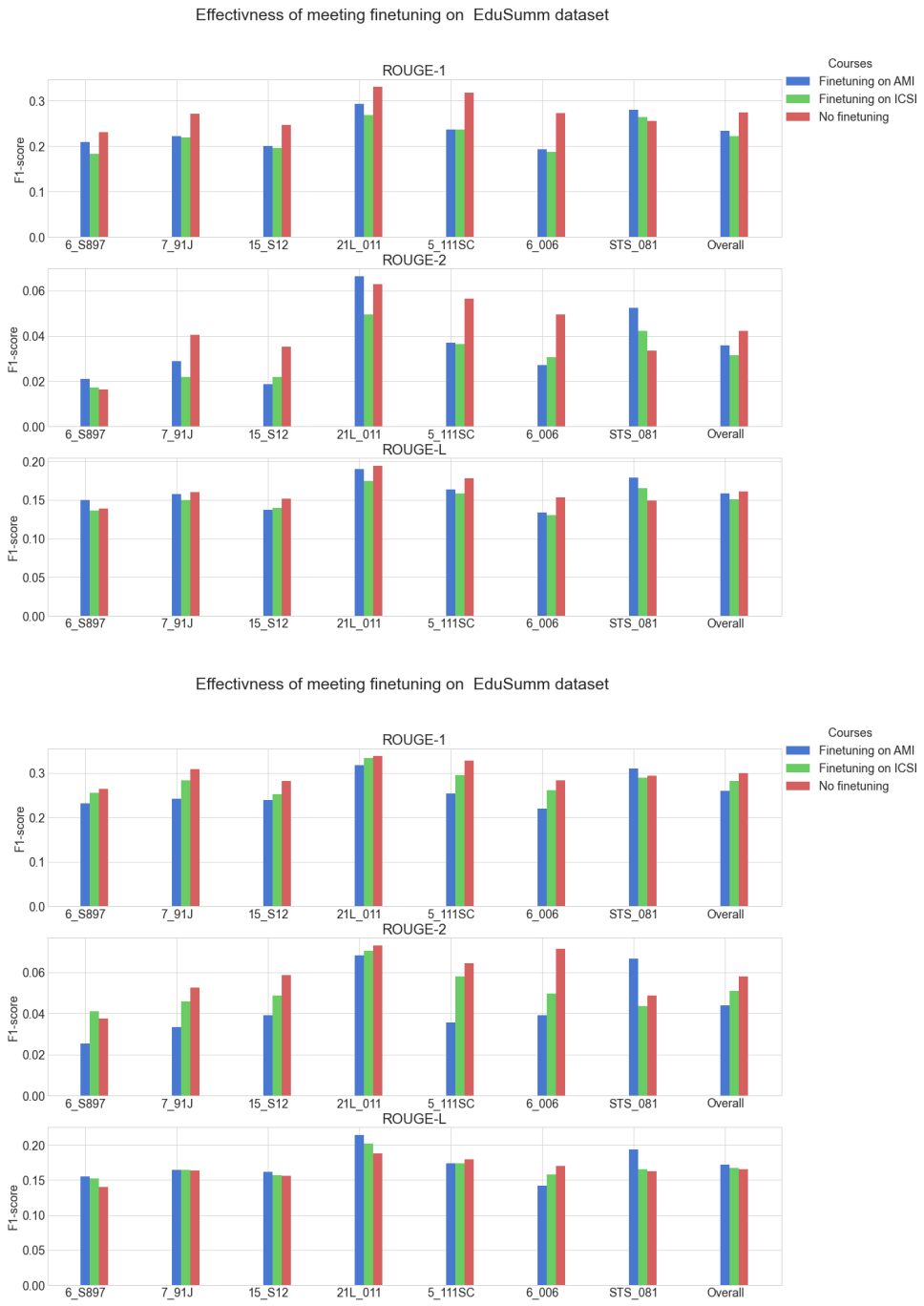

Figure 6.11: Comparison of results obtained with finetuning and without it over the different classes of EduSum with the ROUGE metric. The last columns display the overall results, regardless of the class. The models employed for this comparison are LSA + BART (top) and TextRank + BART (bottom).

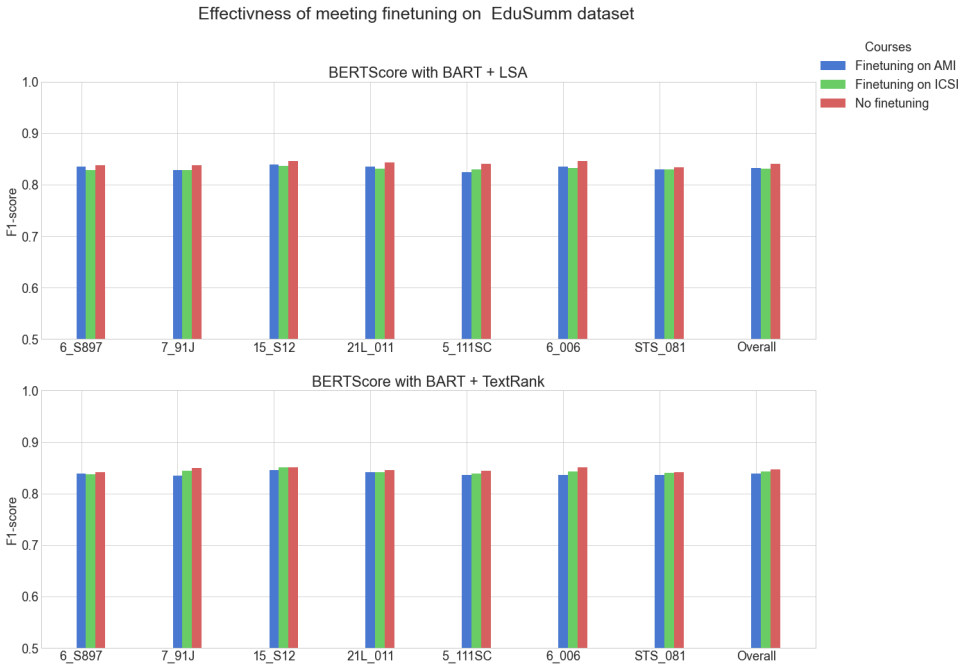

Figure 6.12: Comparison of results obtained with finetuning and without it over the different classes of EduSum dataset with BERTScore. The last columns display the overall results, regardless of the class. The models employed for this comparison are LSA + BART (top) and TextRank + BART (bottom).

6.4.3 Comparison of the two models

This section comments on the benefits of the proposed method in a meeting-to-video lecture domain adaptation setting. In most cases, the presented method demonstrates a higher generalization capability and better adaptability to the educational domain with respect to HMNet.

Both two models provide interesting results in the video lecture domain, but still present some criticalities. From the results described in the previous paragraphs, it is important to notice that the Extractive-Abstractive approach with BART model tends to favor the precision-related measures at the expense of the recall-related ones [2]. Indeed, inspecting the output summaries, it is possible to observe that the generated texts are capable of capturing some important elements of the gold summaries and reproduce them in a very similar manner, but not enough to cover all their topics. In general, the length of BART output summaries is in line with the gold summaries size.

Nevertheless, it is necessary to consider the results obtained on AMI: comparing the results obtained with Extractive-Abstractive on the validation and test set of this dataset to those obtained on each course of EduSum dataset, there is a drop in performance of at most 0.1, 0.03 and 0.07 for ROUGE-1, ROUGE-2 and ROUGE-L respectively. The opposite happens for ICSI, where, most results obtained with the educational data are in line, or even better, than those obtained on ICSI test and validation sets.

With HMNet architecture such drop is much more pronounced: while the model is capable of reaching a ROUGE-1 score 0.40 on AMI or ICSI, on average this result is halved on EduSum, and this also happens for ROUGE-2 and ROUGE-L scores.

With HMNet output summaries are longer than the ones previously generated with BART architecture: on average they contain 230 words [3]. For Microsoft model results, the major contribution is given by the recall term of all ROUGE-n scores. Most likely, this is due to the fact that the longer the summaries are, the higher is the chance to obtain some words appearing in the references, which lets ROUGE-n precision metrics decrease for the benefit of recall counterparts.

Furthermore, as extensively discussed in the previous section, HMNet summaries denote a strong influence given by AMI and ICSI, remarked by the presence of introductory sentences such as “The marketing manager presents [...]”; in the

[2]The ROUGE-recall is given by dividing of the number of overlaps over the length of the reference summary, while the precision counterpart considers the length of the system summary (model prediction) in the denominator.

[3]Both models allow to set a lower and an upper bound for output summary generation, and HMNet summaries frequently reach the upper bound.

educational domain sentences of this kind are meaningless. In this case the training procedure influences HMNet to the point of generating results that have portions of text in common with the fine-tuning datasets. These two phenomena remark HMNet's poor ability to adapt in domains different than the meetings. Most likely, between the two models, BART is preferable: it is capable of generating higher scores with shorter summaries, and thus it demonstrates a greater capability in domain adaptation. As far as BART is concerned, it is possible to remark the importance of the punctuation restoration task even if its benefits are not visible with the ROUGE metrics: the punctuation restoration improves the extractive summary creation upstream and allows to obtained summaries more fluid and pleasant to read.

These results are highlighted in Figure 6.13: in most cases HMNet scores are lower than those of Extractive-Abstractive model, both in term of ROUGE-score and BERTScore. In appendix A.1 the tables summarize all the results reported in the graphs.

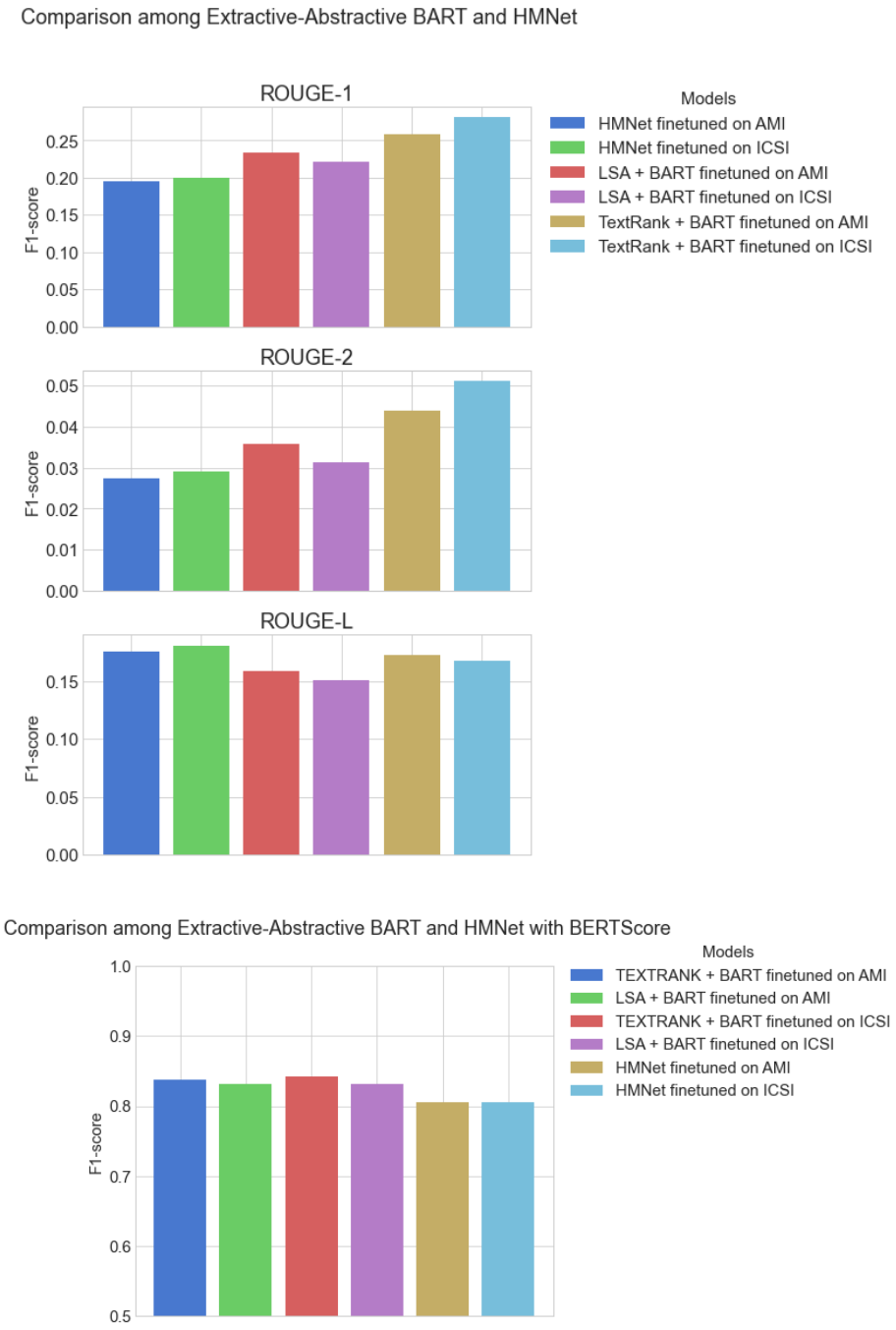

Figure 6.13: Results obtained with EduSum with HMNet finetuned on AMI and on ICSI, evaluated with ROUGE-score (top) and BERTScore (bottom).

6.5 Examples of Politecnico video lecture summary

This section reports some summaries generated with the proposed Extractive-Abstractive model, with different configurations of finetuning and extractive methods.

The following summary is the introductory lecture summary generated from AMI pretrained LSA+BART architecture, which seems to be more coherent with the corresponding lecture:

> *The basic electrical circuits and components need to be considered in order to gain general knowledge about electronics and circuits. In the first part of the course, they are going to talk about voltage and current flows in the opposite direction with respect to the flow of current in a circuit. They will also talk about inductance and magnetic fields, ferromagnetic force, alternating voltages and currents, impedance and transistors, susceptibles and resistors, amplitudes and capacitors and diodes, field less transistors and bipolar junction transistors later on in the course.*

The couterpart obtained with ICSI pretraining seems to be less accurate:

> *The purpose of the project is to help the student in the organization of the program. The cause of the the contract was to control the flow of the current through the circuit, and to limit the number of active and passive components present in the class.The contract was intended to be used to teach the students about the the function of the a series of resistors and to increase the volume of the power produced in the the circuit. However, the contract forced the students to learn about the functions of the A series of the "A series of Heaps".*

Without any finetuning, the result is the following:

> *The method of calculating the voltage difference between two points inside the circuit is based on the text, electronics, the system approach and the structure of the book. It's important to understand the theory of the curse. They will introduce resistors during the course. In the course, they will introduce capacitors and inductors which can be exploited for introducing dc circuits. But in the course with resistors, there are a factor of two from one to the other. The first part of the course is about the definition of duty cycle.*

The results obtained with Text-Rank+BART finetuned on AMI:

There is a problem with the design of the symbol in the following code. The major issues related to the design are the flow of current into and out of a circuit and the function of the power supply in the surrounding areas. The first part of the program is to talk about the properties of the control channel and the functions of the force channel. The second part is to show the different properties of these and to show them how they can be used in a given area.

With TextRank+BART finetuned on ICSI:

The basic electrical circuits and components need to be considered in order to learn about the basic principles of electrical circuits. They will talk about various issues related to electrical circuits, including the flow of current into a circuit and the movement of the current within the circuit. They are going to discuss the major parts of the present course with the rest of the students later on in the course.They will also talk about inductances and transistors, capacitors and diodes, field less transistors and bipolar junction transistors in general and in particular, circuits based on operational amplifiers.

And with the pretrained version that model:

In electrical circuits and components, there are a lot of different kinds of signals that can be used for different parts of the circuit. The first part of the course is related to electrical circuits. It's related to the concept of voltage and current. Next, we will discuss in details about capacitors and diodes, field less transistors, bipolar junction transistors and other types of circuits. In the second part, we're going to discuss in detail about electric currents and voltages and currents.

Chapter 7

Conclusions and final remarks

The goal of this work was to present a possible solution for generating brief descriptions of video lecture contents starting from their transcripts. This task may include a preprocessing step consisting of a punctuation restoration operation, as transcripts derive from an ASR system and might not be punctuated. In addition, video lecture transcripts are the results of spoken language. Cited studies highlighted the risks of using models trained on written text: such models may not guarantee the same performance when applied to transcripts of spoken language. Lastly, video lecture transcriptions are typically longer than news articles, while state-of-the-art models put a constraint on the accepted input length to overcome out-of-memory issues. Given the absence of a dataset for abstractive summarization in this domain, this dissertation explored the possibility of adapting models from meeting to video lecture domain.

In the first chapter, the thesis introduced the problem and discussed the possible applications; in the second chapter this work reviewed some important concepts in the fields of Natural Language Processing. In particular, it investigated the differences between language modelling and sequence labelling and the architectures that are well known in the literature. This distinction was helpful to discriminate models adopted to address all the tasks involved in this work.

The thesis continued with a description of the datasets involved, both for punctuation restoration (IWSLT) and for summarization (AMI and ICSI for abstractive meeting summarization, Politecnico video lecture data). Moreover, it presented a novel dataset, called EduSum, which consists of pairs of transcriptions and abstractive summaries of MIT OpenCourseWare courses. Unlike Politecnico video lecture transcripts, the selected MIT courses transcripts provide the correct punctuation and this allows to assess the impact of punctuation restoration on the

summarization models.

In the fourth chapter state-of-the-art deep learning models in the punctuation restoration task were described. Specifically, a solution for this problem was a RoBERTa-based model that demonstrated good generalization capability even in the video lecture domain. The thesis continued with a description of the state-of-the art models in summarization, analyzing both the extractive and abstractive methods. It gave particular attention to HMNet, a hierarchical structure proposed by Microsoft research group which at the present moment represents the best performing model in meeting summarization. Then, the dissertation examined the concept of Transfer Learning and Domain Adaptation.

Microsoft's architecture, once adequately modified to accomplish this task, revealed to be heavily influenced by the fine-tuning stage on the meetings datasets and has exhibited poor domain adaptation capability. The drop in performance given by the domain shift and the memory requirements of this architecture stimulated the research of alternatives. Therefore, the thesis proposed a novel approach called Extractive-Abstractive, which employs extractive and abstractive algorithms to overcome the previously described limitation. The proposed architecture is formed by an extractive algorithm, which picks the most relevant sentences. The selected portions of text feed the BART model to produce the abstractive summary. The first stage's goal is to create an input text that respects the input size limitation imposed by BART model to avoid memory issues. Only unsupervised extractive algorithms are considered, as they do not need labels for training and do not present any limitation in terms of input length. The second step composes the abstractive summary of the desired length. To improve generalization capabilities and prevent the representational collapse, the abstractive model has been trained with the R3F regularizer.

In chapter six, the thesis discussed about the experiments conducted and the results. In the Extractive-Abstractive approach, the punctuation restoration task seemed to be particularly useful, especially for the generation of the extractive summaries: the resulting summaries are more correct from a grammatical standpoint, even if the improvements in terms of ROUGE scores are limited.

In most cases, the proposed method turned out to be more robust to the domain shift and outperformed the hierarchical architecture in the case of video lecture summarization. With this novel architecture, the advantages are also visible in terms of resources: fine-tuning the entire Extractive-Abstractive architecture has required one single GPU while HMNet needed four of them. The thesis also highlighted improvements in terms of inference time: the time required for generating a new summary is less in the case of the proposed architecture with respect to HMNet as contains fewer attention layers, resulting in 30% fewer parameters. This reduces the complexity and saves resources.

7.1 Future works

The obtained results encourage further exploration of this field. Potential enhancements could be obtained by working on both extractive and abstractive components. It is important to point out that, the extractive summarization methods presented are currently considered as baselines in the literature, and there exist variants that improve their efficiency and results. Moreover, having a good margin in terms of resources, further improvements can be obtained by increasing the size of the model or using the cited alternative versions of BART, which propose variations on the attention mechanism for reducing its complexity. This reasoning on complexity and resources fosters two important remarks.

Firstly, the concept of democratizing artificial intelligence: limiting the size of these large language models could be helpful these new technologies to be more accessible and affordable and to increase their diffusion. The hardware or the premium cloud-based solutions required for training huge deep learning models are still expensive nowadays, and in some cases they are only accessible to large company research centers. However, the possibility of having lighter and faster models could diminish these entry barriers.

Secondly, another aspect to take into account is the energy consumption: in general, it is true that the more parameters the model has, the more computations and memory accesses are required and the higher the energy requirements are [101]. AI models consume a massive amount of energy, and these energy requirements are still growing at a surprising rate. According to [102], on average, the computational resources needed to produce a best-in-class AI model doubled every 3.4 months; this is equivalent to a 300000x increase between 2012 and 2018. Authors in [103] found that the carbon emission costs increase faster than the real improvements gained with larger models and long training.

However, it is important to remark that deploying AI models in real-world settings consumes even more energy than the training does. According to the CEO of Amazon Web Services Andy Jassy [104], about 80% of the cost of a neural network is in inference rather than training. All of these considerations encourage the researchers to keep exploring NLP fields with a multi-objective criterion: on one side, enhance the performance, and on the other, minimize the footprint and the model's negative impacts on the environment.

Appendix A

Tables of results

In this appendix, some tables of results regarding punctuation and summarization task are reported.

A.1 Punctuation restoration results

Results on EduSum dataset					
Course			Punctuator	FastPunct	Transformer
21L-011	Period	Precision	0.180	0.530	0.796
		Recall	0.143	0.391	0.810
		F1-score	0.159	0.444	0.801
	Comma	Precision	0.288	0.481	0.570
		Recall	0.329	0.335	0.746
		F1-score	0.305	0.231	0.643
	Question mark	Precision	0.297	0.140	0.713
		Recall	0.120	0.157	0.744
		F1-score	0.171	0.144	0.724
STS-081	Period	Precision	0.222	0.701	0.548
		Recall	0.162	0.434	0.700
		F1-score	0.187	0.531	0.612
	Comma	Precision	0.247	0.463	0.539
		Recall	0.333	0.316	0.700
		F1-score	0.283	0.360	0.605
	Question mark	Precision	0.345	0.402	0.713
		Recall	0.164	0.299	0.745
		F1-score	0.219	0.341	0.725
5-111SC	Period	Precision	0.222	0.643	0.820
		Recall	0.162	0.369	0.778
		F1-score	0.187	0.463	0.798
	Comma	Precision	0.247	0.228	0.535
		Recall	0.333	0.335	0.677
		F1-score	0.283	0.260	0.590
	Question mark	Precision	0.345	0.532	0.742
		Recall	0.164	0.209	0.802
		F1-score	0.219	0.371	0.768

Table A.1: The table above summarizes the results of punctuation restoration task on EduSum dataset, separately for each class and for each punctuation marks.

Results on EduSum dataset					
Course			Punctuator	FastPunct	Transformer
7-91J	Period	Precision	0.203	0.636	0.809
		Recall	0.172	0.375	0.756
		F1-score	0.186	0.466	0.777
	Comma	Precision	0.281	0.399	0.548
		Recall	0.332	0.29	0.659
		F1-score	0.3	0.349	0.592
	Question mark	Precision	0.221	0.515	0.761
		Recall	0.084	0.305	0.673
		F1-score	0.162	0.434	0.751
6-S897	Period	Precision	0.191	0.659	0.807
		Recall	0.174	0.388	0.762
		F1-score	0.181	0.488	0.78
	Comma	Precision	0.358	0.611	0.628
		Recall	0.394	0.325	0.773
		F1-score	0.371	0.394	0.689
	Question mark	Precision	0.214	0.389	0.701
		Recall	0.124	0.469	0.738
		F1-score	0.159	0.4	0.74
15-S12	Period	Precision	0.297	0.699	0.764
		Recall	0.196	0.459	0.608
		F1-score	0.235	0.549	0.675
	Comma	Precision	0.208	0.402	0.465
		Recall	0.351	0.335	0.68
		F1-score	0.256	0.378	0.571
	Question mark	Precision	0.3	0.452	0.687
		Recall	0.117	0.224	0.699
		F1-score	0.183	0.351	0.678
6-006	Period	Precision	0.204	0.649	0.849
		Recall	0.165	0.352	0.768
		F1-score	0.181	0.449	0.801
	Comma	Precision	0.243	0.327	0.561
		Recall	0.317	0.241	0.676
		F1-score	0.269	0.32	0.616
	Question mark	Precision	0.197	0.559	0.765
		Recall	0.089	0.267	0.643
		F1-score	0.197	0.409	0.675

Table A.2: The table above summarizes the results of punctuation restoration task on EduSum dataset, separately for each class and for each punctuation marks.

A.2 Summarization results: ROUGE score

ROUGE-scores results on EduSum dataset							
Results on 21L-011 dataset with models finetuned on AMI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.343	0.292	0.313	0.083	0.072	0.076
	LSA-BART	0.339	0.268	0.298	0.061	0.05	0.054
	HMNet	0.135	0.401	0.201	0.022	0.064	0.032
No punctuation	TextRank-BART	0.356	0.297	0.321	0.064	0.055	0.059
	LSA-BART	0.344	0.244	0.28	0.069	0.052	0.059
	HMNet	0.14	0.413	0.208	0.022	0.063	0.033
Transformer	TextRank-BART	0.351	0.281	0.307	0.07	0.058	0.063
	LSA-BART	0.335	0.272	0.298	0.054	0.043	0.047
	HMNet	0.141	0.398	0.206	0.02	0.056	0.029
Results on STS-081 dataset with models finetuned on AMI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.261	0.39	0.303	0.05	0.073	0.057
	LSA-BART	0.253	0.368	0.288	0.051	0.075	0.058
	HMNet	0.254	0.367	0.29	0.043	0.061	0.049
No punctuation	TextRank-BART	0.268	0.384	0.302	0.046	0.07	0.053
	LSA-BART	0.263	0.337	0.281	0.045	0.059	0.048
	HMNet	0.254	0.366	0.291	0.044	0.064	0.05
Transformer	TextRank-BART	0.248	0.376	0.286	0.043	0.066	0.05
	LSA-BART	0.249	0.376	0.29	0.038	0.059	0.044
	HMNet	0.261	0.375	0.296	0.044	0.066	0.051
Results on 6-S897 dataset with models finetuned on AMI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.325	0.183	0.232	0.036	0.02	0.026
	LSA-BART	0.288	0.168	0.209	0.028	0.017	0.021
	HMNet	0.128	0.409	0.194	0.017	0.054	0.025
No punctuation	TextRank-BART	0.318	0.179	0.226	0.042	0.025	0.031
	LSA-BART	0.29	0.149	0.191	0.03	0.016	0.021
	HMNet	0.121	0.408	0.185	0.016	0.053	0.024
Transformer	TextRank-BART	0.308	0.179	0.224	0.044	0.024	0.03
	LSA-BART	0.285	0.159	0.202	0.025	0.016	0.019
	HMNet	0.13	0.435	0.198	0.02	0.065	0.03

Results on 7-91J dataset with models finetuned on AMI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.308	0.203	0.242	0.043	0.028	0.034
	LSA-BART	0.302	0.18	0.222	0.039	0.023	0.029
	HMNet	0.117	0.379	0.177	0.012	0.042	0.019
No punctuation	TextRank-BART	0.31	0.205	0.243	0.051	0.031	0.038
	LSA-BART	0.302	0.166	0.206	0.053	0.028	0.035
	HMNet	0.123	0.415	0.188	0.015	0.052	0.022
Transformer	Textrank-BART	0.309	0.194	0.235	0.053	0.033	0.04
	LSA-BART	0.273	0.184	0.216	0.034	0.022	0.026
	HMNet	0.119	0.397	0.181	0.014	0.047	0.021
Results on 6-006 dataset with models finetuned on AMI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.325	0.168	0.219	0.059	0.03	0.039
	LSA-BART	0.302	0.143	0.192	0.043	0.02	0.027
	HMNet	0.095	0.41	0.153	0.013	0.058	0.021
No punctuation	TextRank-BART	0.328	0.162	0.216	0.066	0.031	0.042
	LSA-BART	0.214	0.136	0.156	0.017	0.008	0.011
	HMNet	0.108	0.403	0.167	0.011	0.044	0.018
Transformer	TextRank-BART	0.318	0.145	0.198	0.056	0.025	0.035
	LSA-BART	0.287	0.13	0.178	0.043	0.019	0.027
	HMNet	0.098	0.402	0.155	0.013	0.056	0.021
Results on 15-S12 dataset with models finetuned on AMI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.345	0.188	0.238	0.059	0.03	0.039
	LSA-BART	0.3	0.154	0.2	0.031	0.014	0.019
	HMNet	0.092	0.396	0.148	0.013	0.059	0.021
No punctuation	TextRank-BART	0.314	0.154	0.203	0.06	0.029	0.038
	LSA-BART	0.125	0.123	0.118	0.02	0.03	0.02
	HMNet	0.108	0.375	0.164	0.014	0.048	0.021
Transformer	TextRank-BART	0.36	0.187	0.241	0.074	0.037	0.048
	LSA-BART	0.34	0.163	0.216	0.053	0.023	0.032
	HMNet	0.091	0.379	0.144	0.011	0.049	0.018

Results on 5-111SC dataset with models finetuned on AMI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.311	0.217	0.252	0.055	0.038	0.044
	LSA-BART	0.292	0.196	0.231	0.039	0.027	0.031
	HMNet	0.132	0.419	0.199	0.017	0.052	0.025
No punctuation	TextRank-BART	0.331	0.232	0.269	0.055	0.038	0.044
	LSA-BART	0.144	0.177	0.154	0.009	0.009	0.008
	HMNet	0.143	0.41	0.208	0.018	0.049	0.026
Transformer	TextRank-BART	0.309	0.217	0.252	0.041	0.029	0.034
	LSA-BART	0.287	0.202	0.233	0.042	0.031	0.035
	HMNet	0.136	0.427	0.204	0.017	0.054	0.026

Results on 15-S12 dataset with models finetuned on ICSI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.361	0.201	0.253	0.072	0.038	0.049
	LSA-BART	0.29	0.154	0.197	0.034	0.017	0.022
	HMNet	0.096	0.392	0.152	0.014	0.055	0.022
No punctuation	TextRank-BART	0.345	0.164	0.217	0.053	0.023	0.032
	LSA-BART	0.211	0.136	0.162	0.012	0.007	0.008
	HMNet	0.083	0.358	0.133	0.008	0.035	0.013
Transformer	TextRank-BART	0.397	0.209	0.266	0.093	0.044	0.058
	LSA-BART	0.304	0.152	0.198	0.044	0.02	0.027
	HMNet	0.097	0.403	0.155	0.012	0.054	0.02

Results on 6-006 dataset with models finetuned on ICSI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.396	0.198	0.261	0.077	0.037	0.05
	LSA-BART	0.321	0.133	0.187	0.055	0.021	0.031
	HMNet	0.099	0.442	0.162	0.014	0.065	0.024
No punctuation	TextRank-BART	0.39	0.173	0.237	0.08	0.033	0.046
	LSA-BART	0.26	0.131	0.172	0.02	0.009	0.012
	HMNet	0.106	0.409	0.165	0.012	0.049	0.019
Transformer	TextRank-BART	0.395	0.181	0.245	0.078	0.035	0.048
	LSA-BART	0.304	0.138	0.188	0.041	0.019	0.026
	HMNet	0.099	0.446	0.161	0.013	0.058	0.021

Results on 21L-011 dataset with models finetuned on ICSI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.387	0.281	0.322	0.087	0.064	0.073
	LSA-BART	0.341	0.221	0.266	0.063	0.04	0.04
	HMNet	0.135	0.401	0.201	0.022	0.064	0.032
No punctuation	TextRank-BART	0.428	0.261	0.315	0.098	0.063	0.074
	LSA-BART	0.363	0.212	0.262	0.068	0.039	0.048
	HMNet	0.14	0.413	0.208	0.022	0.063	0.033
Transformer	TextRank-BART	0.346	0.306	0.322	0.048	0.045	0.046
	LSA-BART	0.361	0.222	0.269	0.072	0.044	0.054
	HMNet	0.141	0.398	0.206	0.02	0.056	0.029
Results on STS-081 dataset with models finetuned on ICSI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.264	0.369	0.292	0.043	0.06	0.048
	LSA-BART	0.237	0.348	0.269	0.037	0.058	0.043
	HMNet	0.254	0.367	0.29	0.043	0.061	0.049
No punctuation	TextRank-BART	0.277	0.339	0.294	0.041	0.053	0.045
	LSA-BART	0.283	0.307	0.281	0.045	0.049	0.044
	HMNet	0.254	0.366	0.291	0.044	0.064	0.05
Transformer	Textrank-BART	0.272	0.352	0.296	0.043	0.06	0.048
	LSA-BART	0.243	0.346	0.278	0.041	0.059	0.047
	HMNet	0.261	0.375	0.296	0.044	0.066	0.051
Results on 5-111SC dataset with models finetuned on ICSI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.374	0.249	0.293	0.074	0.047	0.056
	LSA-BART	0.34	0.196	0.244	0.062	0.033	0.042
	HMNet	0.132	0.419	0.199	0.017	0.052	0.025
No punctuation	TextRank-BART	0.387	0.229	0.282	0.087	0.049	0.062
	LSA-BART	0.205	0.161	0.177	0.013	0.009	0.01
	HMNet	0.143	0.41	0.208	0.018	0.049	0.026
Transformer	TextRank-BART	0.358	0.259	0.296	0.059	0.044	0.05
	LSA-BART	0.313	0.201	0.239	0.042	0.028	0.033
	HMNet	0.136	0.427	0.204	0.017	0.054	0.026

Results on 6-S897 dataset with models finetuned on ICSI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.36	0.202	0.254	0.054	0.034	0.041
	LSA-BART	0.286	0.138	0.183	0.026	0.013	0.017
	HMNet	0.124	0.419	0.19	0.017	0.055	0.025
No punctuation	TextRank-BART	0.35	0.17	0.222	0.052	0.025	0.033
	LSA-BART	0.282	0.136	0.179	0.027	0.013	0.018
	HMNet	0.125	0.419	0.19	0.015	0.05	0.023
Transformer	TextRank-BART	0.371	0.185	0.242	0.055	0.027	0.035
	LSA-BART	0.303	0.156	0.202	0.031	0.017	0.021
	HMNet	0.128	0.424	0.195	0.017	0.056	0.026
Results on 7-91J dataset with models finetuned on ICSI							
Version	Model	ROUGE-1			ROUGE-2		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.356	0.24	0.283	0.057	0.039	0.046
	LSA-BART	0.305	0.177	0.22	0.03	0.018	0.022
	HMNet	0.114	0.402	0.177	0.015	0.057	0.023
No punctuation	TextRank-BART	0.415	0.216	0.278	0.076	0.038	0.05
	LSA-BART	0.325	0.159	0.207	0.047	0.021	0.028
	HMNet	0.116	0.413	0.18	0.013	0.05	0.021
Transformer	Textrank-BART	0.386	0.209	0.266	0.063	0.033	0.043
	LSA-BART	0.29	0.171	0.211	0.042	0.025	0.03
	HMNet	0.12	0.416	0.185	0.014	0.053	0.022

Table A.3: Evaluation of the impact of punctuation restoration models (column *Version*) on the summarization task: results of summarization task on EduSum dataset, with models pretrained on AMI and ICSI, in terms of ROUGE-1 and ROUGE-2.

A.3 Summarization results: BERTScore

BERTScore results on EduSum dataset							
Results on 21L-011							
Version	Model	Finetuning on AMI			Finetuning on ICSI		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.846	0.837	0.841	0.842	0.842	0.842
	LSA-BART	0.836	0.833	0.834	0.83	0.833	0.831
	HMNet	0.795	0.818	0.806	0.795	0.82	0.807
No punctuation	TextRank-BART	0.84	0.836	0.838	0.841	0.841	0.841
	LSA-BART	0.834	0.832	0.833	0.823	0.829	0.826
	HMNet	0.793	0.818	0.805	0.792	0.818	0.805
Transformer	TextRank-BART	0.842	0.834	0.838	0.842	0.84	0.841
	LSA-BART	0.841	0.834	0.837	0.833	0.832	0.832
	HMNet	0.793	0.817	0.805	0.795	0.82	0.807
Results on STS-081							
Version	Model	Finetuning on AMI			Finetuning on ICSI		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.846	0.825	0.835	0.849	0.831	0.84
	LSA-BART	0.839	0.82	0.829	0.836	0.822	0.829
	HMNet	0.795	0.812	0.803	0.8	0.814	0.807
No punctuation	TextRank-BART	0.846	0.824	0.835	0.842	0.827	0.835
	LSA-BART	0.837	0.82	0.829	0.824	0.82	0.822
	HMNet	0.801	0.813	0.807	0.801	0.815	0.808
Transformer	TextRank-BART	0.85	0.825	0.837	0.848	0.831	0.839
	LSA-BART	0.842	0.822	0.832	0.84	0.823	0.831
	HMNet	0.796	0.812	0.804	0.8	0.815	0.807
Results on 6-S897							
Version	Model	Finetuning on AMI			Finetuning on ICSI		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.839	0.839	0.839	0.833	0.843	0.837
	LSA-BART	0.834	0.836	0.835	0.823	0.835	0.829
	HMNet	0.795	0.825	0.81	0.791	0.824	0.807
No punctuation	TextRank-BART	0.834	0.839	0.836	0.828	0.841	0.834
	LSA-BART	0.824	0.832	0.828	0.814	0.834	0.824
	HMNet	0.798	0.826	0.811	0.793	0.823	0.808
Transformer	TextRank-BART	0.836	0.836	0.836	0.833	0.842	0.837
	LSA-BART	0.833	0.835	0.834	0.816	0.835	0.825
	HMNet	0.792	0.824	0.808	0.789	0.824	0.806

Results on 7-91J dataset with models finetuned on AMI							
Version	Model	Finetuning on AMI			Finetuning on ICSI		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.832	0.839	0.835	0.839	0.848	0.843
	LSA-BART	0.824	0.833	0.829	0.824	0.834	0.829
	HMNet	0.791	0.821	0.806	0.79	0.823	0.806
No punctuation	TextRank-BART	0.833	0.837	0.835	0.832	0.847	0.84
	LSA-BART	0.817	0.834	0.825	0.807	0.83	0.818
	HMNet	0.794	0.823	0.808	0.791	0.822	0.806
Transformer	Textrank-BART	0.834	0.84	0.837	0.84	0.849	0.844
	LSA-BART	0.83	0.832	0.831	0.823	0.832	0.827
	HMNet	0.79	0.821	0.805	0.788	0.822	0.805
Results on 6-006 dataset with models							
Version	Model	Finetuning on AMI			Finetuning on ICSI		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.83	0.843	0.836	0.832	0.853	0.843
	LSA-BART	0.827	0.842	0.834	0.823	0.843	0.833
	HMNet	0.788	0.825	0.806	0.786	0.827	0.806
No punctuation	TextRank-BART	0.832	0.847	0.839	0.829	0.854	0.841
	LSA-BART	0.788	0.825	0.806	0.814	0.826	0.82
	HMNet	0.795	0.827	0.811	0.788	0.824	0.806
Transformer	TextRank-BART	0.829	0.844	0.836	0.83	0.851	0.84
	LSA-BART	0.824	0.839	0.831	0.817	0.839	0.828
	HMNet	0.79	0.826	0.808	0.785	0.827	0.805
Results on 15-S12 dataset with models							
Version	Model	Finetuning on AMI			Finetuning on ICSI		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.842	0.849	0.845	0.844	0.857	0.85
	LSA-BART	0.837	0.842	0.839	0.83	0.843	0.836
	HMNet	0.791	0.826	0.808	0.787	0.825	0.806
No punctuation	TextRank-BART	0.839	0.845	0.842	0.826	0.845	0.835
	LSA-BART	0.767	0.821	0.793	0.807	0.826	0.816
	HMNet	0.799	0.824	0.811	0.786	0.821	0.803
Transformer	TextRank-BART	0.846	0.851	0.848	0.843	0.855	0.849
	LSA-BART	0.834	0.842	0.838	0.824	0.842	0.833
	HMNet	0.793	0.826	0.809	0.788	0.826	0.807

Results on 5-111SC							
Version	Model	Finetuning on AMI			Finetuning on ICSI		
		P	R	F1	P	R	F1
Original	TextRank-BART	0.834	0.837	0.835	0.833	0.844	0.839
	LSA-BART	0.816	0.833	0.824	0.826	0.834	0.83
	HMNet	0.787	0.815	0.801	0.787	0.814	0.8
No punctuation	TextRank-BART	0.833	0.836	0.834	0.831	0.844	0.837
	LSA-BART	0.772	0.81	0.791	0.806	0.813	0.809
	HMNet	0.79	0.812	0.801	0.788	0.81	0.799
Transformer	TextRank-BART	0.833	0.837	0.835	0.832	0.842	0.837
	LSA-BART	0.823	0.833	0.828	0.828	0.833	0.83
	HMNet	0.786	0.814	0.8	0.788	0.815	0.801

Table A.4: BERTScore results of summarization task on EduSum dataset, with models pretrained on AMI and ICSI.

Appendix B

Additional notes

B.1 Residual connections

Authors in [105] discover that when the network depth increases, accuracy gets saturated and then degrades. According to them, this is not due to overfitting because they reported degradation even in training accuracy. To address this issue the authors proposed the *deep residual learning framework*. Residual connection uses skip connections to avoid some layers: given an input $\boldsymbol{x}$ of a layer, $\boldsymbol{y}$ the output of such layer, $\mathcal{F}(\cdot)$ an underlying residual mapping to learn, the building block of these connections are given by:

$$\boldsymbol{y} = \mathcal{F}(\boldsymbol{x}, \{W_i\}) + \boldsymbol{x} \tag{B.1}$$

$\mathcal{F}(\cdot)$ represents a non-linear transformation of the input given by learnable parameters W_i. This way, it is possible to increase the number of layers and avoiding the vanishing gradient problem [1]. Residual networks are networks that adopt skip connection, and in particular *ResNet* became a popular backbone in Computer vision application. This network beat the concurrence of SOTA models in ImageNet challenge in 2015 (ILSVRC 2015 [106] - image classification).

The idea of the skip connections derives from the Highway networks [107] which present shortcut connections with gate functions (so parametrized). Residual connections are Highway networks block whose gates are always open. Image in B.1 shows the building block of a residual network.

[1] This problem typically involves the first parameters of long network: during the backpropagation, the gradients of the first layers vanishes and the parameters update are irrelevant. Controlling the gradient flows means obtaining a model whose first layers are able to learn as the last ones.

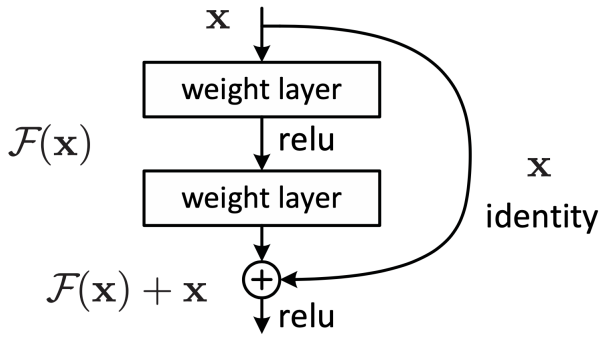

Figure B.1: Resudual connection schema presents in [105]

B.2 Adam and AdamW optimizers

In [96] the authors propose an alternative to the SGD optimizer [108] capable of computing individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients.

The Adam considers the optimization function as stochastic and seeks to minimize the expected value of this funcion, $\mathbb{E}[f(\theta)]$ with respect to the parameters θ. The algorithm updates the estimates of the 1^{st} moment, m_t and the second raw moment v_t of the grandient, by updating their eponential mooving average. This introduces two hyperparameters, β_1, $\beta_2 \in [0, 1)$ that control the exponential decay rates of these moving averages. The fist moment estimate is equal to:

$$m_t = (1 - \beta_1) \sum_{i=0}^{t} \beta_1^{t-1} g_i \tag{B.2}$$

and the second moment is given by:

$$v_t = (1 - \beta_2) \sum_{i=1}^{t} \beta_2^{t-1} \cdot g_i^2 \tag{B.3}$$

which both are biased estimates; the unbiased versions [2] can be obtaining via:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{B.4}$$

and

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{B.5}$$

[2] An estimator is considered unbiased if its expected value converges to the expected value of what it esteems.

and the update is of the parameters is given by:

$$\Delta_t = \alpha \cdot \hat{m}_t / \hat{v}_t \tag{B.6}$$

The step size of Adam update rule is invariant to the magnitude of the gradient, and this helps in the case of vanishing gradient problem, where SGD would be stuck in a local minimum. Furthermore, its step sizes are approximately bounded by the step size hyperparameters, which, with this interpretation, can be tuned easily. The complete algorithm is showed in Figure B.2.

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

Figure B.2: Adam optimization algorithm pseudocode presented in [96]

AdamW, differs from Adam for different aspects. The authors in [95] point out that that L_2 regularization and weight decay regularization are equivalent in the case of standard stochastic gradient descent algorithm, but not in the case of adaptive gradient algorithms like Adam. Also, they stated that L_2 regularization is not effective in Adam, since the regularizer gradient gets scaled along the loss gradient, and weights, in case the training generates large gradients in the loss function, do not get regularized as much as they would. For this reason, they propose AdamW a version of Adam that explicitly decouples λ and α for the weight update.

Instead of adding the regularization term in the loss function, as in standard Adam, the regularizer is intruduced only when the weights are updated. This way,

the gradient of regularization terms is not rescaled by the gradient itself and can be controlled by precise parameters. The update formula becomes as follows:

$$\theta_t = \theta_{t-1} - \eta_t \left(\alpha \cdot \frac{\hat{m}_t}{(\sqrt{\hat{v}_t} + \epsilon)} + \lambda \theta_{t-1} \right) \tag{B.7}$$

The pseudocode of AdamW is described in Figure B.3.

Algorithm 2 Adam with L_2 regularization and Adam with decoupled weight decay (AdamW)

given $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}, \lambda \in \mathbb{R}$
initialize time step $t \leftarrow 0$, parameter vector $\boldsymbol{\theta}_{t=0} \in \mathbb{R}^n$, first moment vector $\boldsymbol{m}_{t=0} \leftarrow \boldsymbol{0}$, second moment vector $\boldsymbol{v}_{t=0} \leftarrow \boldsymbol{0}$, schedule multiplier $\eta_{t=0} \in \mathbb{R}$
repeat
  $t \leftarrow t + 1$
  $\nabla f_t(\boldsymbol{\theta}_{t-1}) \leftarrow \text{SelectBatch}(\boldsymbol{\theta}_{t-1})$ ▷ select batch and return the corresponding gradient
  $\boldsymbol{g}_t \leftarrow \nabla f_t(\boldsymbol{\theta}_{t-1}) + \lambda \boldsymbol{\theta}_{t-1}$
  $\boldsymbol{m}_t \leftarrow \beta_1 \boldsymbol{m}_{t-1} + (1 - \beta_1) \boldsymbol{g}_t$ ▷ here and below all operations are element-wise
  $\boldsymbol{v}_t \leftarrow \beta_2 \boldsymbol{v}_{t-1} + (1 - \beta_2) \boldsymbol{g}_t^2$
  $\hat{\boldsymbol{m}}_t \leftarrow \boldsymbol{m}_t / (1 - \beta_1^t)$ ▷ β_1 is taken to the power of t
  $\hat{\boldsymbol{v}}_t \leftarrow \boldsymbol{v}_t / (1 - \beta_2^t)$ ▷ β_2 is taken to the power of t
  $\eta_t \leftarrow \text{SetScheduleMultiplier}(t)$ ▷ can be fixed, decay, or also be used for warm restarts
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \left(\alpha \hat{\boldsymbol{m}}_t / (\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon) + \lambda \boldsymbol{\theta}_{t-1} \right)$
until *stopping criterion is met*
return optimized parameters $\boldsymbol{\theta}_t$

Figure B.3: AdamW optimization algorithm pseudocode presented in [95]

Bibliography

[1] Joi L. Moore, Camille Dickson-Deane, and Krista Galyen. «e-Learning, online learning, and distance learning environments: Are they the same?» In: *The Internet and Higher Education* 14.2 (2011). Web mining and higher education: Introduction to the special issue, pp. 129–135. ISSN: 1096-7516. DOI: https://doi.org/10.1016/j.iheduc.2010.10.001. URL: https://www.sciencedirect.com/science/article/pii/S1096751610000886 (cit. on p. 1).

[2] Kimberly C. Harper, Kuanchin Chen, and David C. Yen. «Distance learning, virtual classrooms, and teaching pedagogy in the Internet environment». In: *Technology in Society* 26.4 (2004), pp. 585–598. ISSN: 0160-791X. DOI: https://doi.org/10.1016/j.techsoc.2004.08.002. URL: https://www.sciencedirect.com/science/article/pii/S0160791X04000545 (cit. on p. 1).

[3] Adéle Bezuidenhout. «Implications for academic workload of the changing role of distance educators». In: *Distance Education* 36.2 (2015), pp. 246–262. DOI: 10.1080/01587919.2015.1055055. eprint: https://doi.org/10.1080/01587919.2015.1055055. URL: https://doi.org/10.1080/01587919.2015.1055055 (cit. on p. 1).

[4] Ogechi Ohadomere and Ikedinachi K. Ogamba. «Management-led interventions for workplace stress and mental health of academic staff in higher education: a systematic review». en. In: *The Journal of Mental Health Training, Education and Practice* 16.1 (Jan. 2021), pp. 67–82. ISSN: 1755-6228, 1755-6228. DOI: 10.1108/JMHTEP-07-2020-0048. URL: https://www.emerald.com/insight/content/doi/10.1108/JMHTEP-07-2020-0048/full/html (visited on 06/14/2021) (cit. on p. 1).

[5] Ido Roll and Ruth Wylie. «Evolution and Revolution in Artificial Intelligence in Education». en. In: *International Journal of Artificial Intelligence in Education* 26.2 (June 2016), pp. 582–599. ISSN: 1560-4292, 1560-4306. DOI: 10.1007/s40593-016-0110-3. URL: http://link.springer.com/10.1007/s40593-016-0110-3 (visited on 06/14/2021) (cit. on p. 1).

[6] Xieling Chen, Haoran Xie, and Gwo-Jen Hwang. «A multi-perspective study on Artificial Intelligence in Education: grants, conferences, journals, software tools, institutions, and researchers». In: *Computers and Education: Artificial Intelligence* 1 (2020), p. 100005. ISSN: 2666-920X. DOI: https://doi.org/10.1016/j.caeai.2020.100005. URL: https://www.sciencedirect.com/science/article/pii/S2666920X20300059 (cit. on p. 1).

[7] Olaf Zawacki-Richter, Victoria I. Marín, Melissa Bond, and Franziska Gouverneur. «Systematic review of research on artificial intelligence applications in higher education – where are the educators?» en. In: *International Journal of Educational Technology in Higher Education* 16.1 (Dec. 2019), p. 39. ISSN: 2365-9440. DOI: 10.1186/s41239-019-0171-0. URL: https://educationaltechnologyjournal.springeropen.com/articles/10.1186/s41239-019-0171-0 (visited on 06/14/2021) (cit. on p. 1).

[8] Adina Magda Florea and Serban Radu. «Artificial Intelligence and Education». In: *2019 22nd International Conference on Control Systems and Computer Science (CSCS)*. 2019, pp. 381–382. DOI: 10.1109/CSCS.2019.00069 (cit. on p. 2).

[9] Elena Baralis and Luca Cagliero. «Learning From Summaries: Supporting e-Learning Activities by Means of Document Summarization». In: *IEEE Transactions on Emerging Topics in Computing* 4.3 (2016), pp. 416–428. DOI: 10.1109/TETC.2015.2493338 (cit. on p. 2).

[10] Luca Cagliero, Laura Farinetti, and Elena Baralis. «Recommending Personalized Summaries of Teaching Materials». In: *IEEE Access* 7 (2019), pp. 22729–22739. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2899655. URL: https://ieeexplore.ieee.org/document/8642858/ (visited on 06/16/2021) (cit. on pp. 2, 45).

[11] Agus Wedi, Saida Ulfa, Ashar Pakkawaru, and Rex Bringula. «Exploring the Implementation of Automatic Text Summarization in Online Learning Setting». In: *Proceedings of the 1 st International Conference on Information Technology and Education (ICITE 2020)*. Atlantis Press, 2020, pp. 5–8. ISBN: 978-94-6239-299-1. DOI: https://doi.org/10.2991/assehr.k.201214.203. URL: https://doi.org/10.2991/assehr.k.201214.203 (cit. on pp. 2, 45).

[12] Derek Miller. «Leveraging BERT for Extractive Text Summarization on Lectures». In: *CoRR* abs/1906.04165 (2019). arXiv: 1906.04165. URL: http://arxiv.org/abs/1906.04165 (cit. on pp. 2, 46).

[13] *LAK '15: Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*. Poughkeepsie, New York: Association for Computing Machinery, 2015. ISBN: 9781450334174 (cit. on p. 2).

[14] Sansiri Tarnpradab, Fei Liu, and Kien A. Hua. *Toward Extractive Summarization of Online Forum Discussions via Hierarchical Attention Networks*. 2018. arXiv: 1805.10390 [cs.CL] (cit. on p. 2).

[15] Swapna Gottipati, Venky Shankararaman, and Renjini Ramesh. «TopicSummary: A Tool for Analyzing Class Discussion Forums using Topic Based Summarizations». In: *2019 IEEE Frontiers in Education Conference (FIE)*. Covington, KY, USA: IEEE, Oct. 2019, pp. 1–9. ISBN: 9781728117461. DOI: 10.1109/FIE43999.2019.9028526. URL: https://ieeexplore.ieee.org/document/9028526/ (visited on 06/16/2021) (cit. on pp. 2, 42, 45).

[16] Atsushi Shimada, Fumiya Okubo, Chengjiu Yin, and Hiroaki Ogata. «Automatic Summarization of Lecture Slides for Enhanced Student Preview-Technical Report and User Study». English. In: *IEEE Transactions on Learning Technologies* 11.2 (Apr. 2018). Funding Information: This research was partially supported by "PRESTO", Japan Science and Technology Agency (JST) Japan, and "Research and Development on Fundamental and Utilization Technologies for Social Big Data" (178A03), the Commissioned Research of the National Institute of Information and Communications Technology (NICT) Japan. Publisher Copyright: © 2008-2011 IEEE., pp. 165–178. ISSN: 1939-1382. DOI: 10.1109/TLT.2017.2682086 (cit. on pp. 2, 45).

[17] Shruti Garg. «Automatic Text Summarization of Video Lectures Using Subtitles». In: *Recent Developments in Intelligent Computing, Communication and Devices*. Ed. by Srikanta Patnaik and Florin Popentiu-Vladicescu. Singapore: Springer Singapore, 2017, pp. 45–52. ISBN: 978-981-10-3779-5 (cit. on pp. 2, 46).

[18] Jacob Eisenstein. *Natural Language Processing*. MIT Press, 2018 (cit. on pp. 5, 11).

[19] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. «Efficient Estimation of Word Representations in Vector Space». In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2013. URL: http://arxiv.org/abs/1301.3781 (cit. on pp. 6, 8).

[20] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. «A Neural Probabilistic Language Model». In: *J. Mach. Learn. Res.* 3.null (Mar. 2003), pp. 1137–1155. ISSN: 1532-4435 (cit. on p. 6).

[21] Tomás Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. «Distributed Representations of Words and Phrases and their Compositionality». In: *CoRR* abs/1310.4546 (2013). arXiv: 1310.4546. URL: http://arxiv.org/abs/1310.4546 (cit. on p. 7).

[22] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. «GloVe: Global Vectors for Word Representation». In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162 (cit. on p. 7).

[23] D. E. Rumelhart and J. L. McClelland. «Learning Internal Representations by Error Propagation». In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*. 1987, pp. 318–362 (cit. on p. 10).

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL] (cit. on pp. 12, 14, 77, 80).

[25] Thomas Wolf et al. «Transformers: State-of-the-Art Natural Language Processing». In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6 (cit. on p. 13).

[26] A. Radford and Karthik Narasimhan. «Improving Language Understanding by Generative Pre-Training». In: 2018 (cit. on pp. 13, 15, 56).

[27] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL] (cit. on pp. 13, 14, 56).

[28] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. arXiv: 1910.13461 [cs.CL] (cit. on pp. 13, 44, 56–58).

[29] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. «Language Models are Unsupervised Multitask Learners». In: (2019) (cit. on p. 15).

[30] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL] (cit. on p. 16).

[31] Yoav Freund and Robert Schapire. «Large Margin Classification Using the Perceptron Algorithm». In: *Machine Learning* 37 (Feb. 1999). DOI: 10.1023/A:1007662407062 (cit. on p. 17).

[32] Corinna Cortes and Vladimir Vapnik. «Support-vector networks». In: *Chem. Biol. Drug Des.* 297 (Jan. 2009), pp. 273–297. DOI: 10.1007/\%2FBF00994018 (cit. on p. 17).

[33] *2011 International Workshop on Spoken Language Translation, IWSLT 2011, San Francisco, CA, USA, December 8-9, 2011*. ISCA, 2011. URL: http://www.isca-speech.org/archive/iwslt%5C_11/ (cit. on pp. 20, 21).

[34] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. «Teaching Machines to Read and Comprehend». In: *NIPS*. 2015 (cit. on pp. 20, 51).

[35] Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. «SAMSum Corpus: A Human-annotated Dialogue Dataset for Abstractive Summarization». In: *CoRR* abs/1911.12237 (2019). arXiv: 1911.12237. URL: http://arxiv.org/abs/1911.12237 (cit. on pp. 20, 58).

[36] Jean Carletta et al. «The AMI meeting corpus: A pre-announcement». In: July 2005. ISBN: 978-3-540-32549-9. DOI: 10.1007/11677482_3 (cit. on p. 20).

[37] A. Janin et al. «The ICSI meeting corpus». In: May 2003, pp. I–364. ISBN: 0-7803-7663-3. DOI: 10.1109/ICASSP.2003.1198793 (cit. on pp. 20, 22).

[38] Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Jean-Pierre Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. «Unsupervised Abstractive Meeting Summarization with Multi-Sentence Compression and Budgeted Submodular Maximization». In: *CoRR* abs/1805.05271 (2018). arXiv: 1805.05271. URL: http://arxiv.org/abs/1805.05271 (cit. on p. 23).

[39] *Innovation Systems for Science, Technology, Energy, Manufacturing, and Health*. en. URL: https://ocw.mit.edu/courses/science-technology-and-society/sts-081-innovation-systems-for-science-technology-energy-manufacturing-and-health-spring-2017/ (cit. on p. 23).

[40] *The Film Experience*. en. URL: https://ocw.mit.edu/courses/literature/21l-011-the-film-experience-fall-2013/ (cit. on p. 23).

[41] *Principles of Chemical Science*. en. URL: https://ocw.mit.edu/courses/chemistry/5-111sc-principles-of-chemical-science-fall-2014/ (cit. on p. 23).

[42] *Introduction to Algorithms*. en. URL: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-006-introduction-to-algorithms-fall-2011/ (cit. on p. 24).

[43] *Machine Learning for Healthcare*. en. URL: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-s897-machine-learning-for-healthcare-spring-2019/ (cit. on p. 24).

[44] *Foundations of Computational and Systems Biology*. en. URL: https://ocw.mit.edu/courses/biology/7-91j-foundations-of-computational-and-systems-biology-spring-2014/ (cit. on p. 24).

[45] *Blockchain and Money*. en. URL: https://ocw.mit.edu/courses/sloan-school-of-management/15-s12-blockchain-and-money-fall-2018/ (cit. on p. 24).

[46] Joshua Y. Kim, Chunfeng Liu, Rafael A. Calvo, Kathryn McCabe, Silas C. R. Taylor, Björn W. Schuller, and Kaihang Wu. «A Comparison of Online Automatic Speech Recognition Systems and the Nonverbal Responses to Unintelligible Speech». In: *CoRR* abs/1904.12403 (2019). arXiv: 1904.12403. URL: http://arxiv.org/abs/1904.12403 (cit. on p. 26).

[47] H. Liao, E. McDermott, and A. Senior. «Large scale deep neural network acoustic modeling with semi-supervised training data for YouTube video transcription». In: *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. 2013, pp. 368–373. DOI: 10.1109/ASRU.2013.6707758 (cit. on p. 26).

[48] Ottokar Tilk and Tanel Alumäe. «Bidirectional Recurrent Neural Network with Attention Mechanism for Punctuation Restoration». In: *INTERSPEECH*. 2016 (cit. on pp. 28, 31, 32, 67, 77, 79).

[49] Agustín Gravano, Martin Jansche, and Michiel Bacchiani. «Restoring Punctuation and Capitalization in Transcribed Speech». In: *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. 2009, pp. 4741–4744 (cit. on pp. 28, 29).

[50] Wei Lu and Hwee Ng. «Better Punctuation Prediction with Dynamic Conditional Random Fields.» In: Jan. 2010, pp. 177–186 (cit. on pp. 28, 30).

[51] Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. «Large Language Models in Machine Translation». In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. 2007, pp. 858–867 (cit. on p. 29).

[52] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL] (cit. on pp. 31, 33, 43).

[53] F. Wang, W. Chen, Zhen Yang, and Bo Xu. «Self-Attention Based Network for Punctuation Restoration». In: *2018 24th International Conference on Pattern Recognition (ICPR)* (2018), pp. 2803–2808 (cit. on pp. 33, 34).

[54] Tanvirul Alam, Akib Khan, and Firoj Alam. «Punctuation Restoration using Transformer Models for High-and Low-Resource Languages». In: *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 132–142. DOI: 10.18653/v1/2020.wnut-1.18. URL: https://www.aclweb.org/anthology/2020.wnut-1.18 (cit. on p. 34).

[55] Yinhan Liu et al. «RoBERTa: A Robustly Optimized BERT Pretraining Approach». In: *CoRR* abs/1907.11692 (2019). arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692 (cit. on p. 35).

[56] Dragomir R. Radev, E. Hovy, and K. McKeown. «Introduction to the Special Issue on Summarization». In: *Computational Linguistics* 28 (2002), pp. 399–408 (cit. on p. 36).

[57] Dipanjan Das and André F. T. Martins. *A Survey on Automatic Text Summarization*. 2007 (cit. on p. 36).

[58] Julian Kupiec, Jan Pedersen, and Francine Chen. «A Trainable Document Summarizer». In: *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '95. Seattle, Washington, USA: Association for Computing Machinery, 1995, pp. 68–73. ISBN: 0897917146. DOI: 10.1145/215206.215333. URL: https://doi.org/10.1145/215206.215333 (cit. on p. 37).

[59] T. Landauer and S. Dumais. «A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge.» In: *Psychological Review* 104 (1997), pp. 211–240 (cit. on p. 37).

[60] Thomas K Landauer, Peter W. Foltz, and Darrell Laham. «An introduction to latent semantic analysis». In: *Discourse Processes* 25.2-3 (1998), pp. 259–284. DOI: 10.1080/01638539809545028. eprint: https://doi.org/10.1080/01638539809545028. URL: https://doi.org/10.1080/01638539809545028 (cit. on p. 37).

[61] Makbule Ozsoy, Ferda Alpaslan, and Ilyas Cicekli. «Text summarization using Latent Semantic Analysis». In: *J. Information Science* 37 (Aug. 2011), pp. 405–417. DOI: 10.1177/0165551511408848 (cit. on p. 39).

[62] Yihong Gong and Xin Liu. «Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis». In: *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '01. New Orleans, Louisiana, USA: Association for Computing Machinery, 2001, pp. 19–25. ISBN: 1581133316. DOI: 10.1145/383952.383955. URL: https://doi.org/10.1145/383952.383955 (cit. on p. 39).

[63] Josef Steinberger and Karel Jezek. «Using Latent Semantic Analysis in Text Summarization and Summary Evaluation». In: Jan. 2004 (cit. on p. 39).

[64] Makbule Gulcin Ozsoy, Ilyas Cicekli, and Ferda Nur Alpaslan. «Text Summarization of Turkish Texts Using Latent Semantic Analysis». In: *Proceedings of the 23rd International Conference on Computational Linguistics*. COLING '10. Beijing, China: Association for Computational Linguistics, 2010, pp. 869–876 (cit. on p. 40).

[65] R. Mihalcea and P. Tarau. «TextRank: Bringing Order into Text». In: *EMNLP*. 2004 (cit. on pp. 40, 90).

[66] Sergey Brin and Lawrence Page. «The anatomy of a large-scale hypertextual Web search engine». In: *Computer Networks and ISDN Systems* 30.1 (1998). Proceedings of the Seventh International World Wide Web Conference, pp. 107–117. ISSN: 0169-7552. DOI: https://doi.org/10.1016/S0169-7552(98)00110-X. URL: https://www.sciencedirect.com/science/article/pii/S016975529800110X (cit. on p. 40).

[67] Wael Gomaa and Aly Fahmy. «A Survey of Text Similarity Approaches». In: *international journal of Computer Applications* 68 (Apr. 2013). DOI: 10.5120/11638-7118 (cit. on p. 41).

[68] Vasileios Hatzivassiloglou, Judith Klavans, Melissa Holcombe, Regina Barzilay, Min-yen Kan, and Kathleen McKeown. «SIMFINDER: A flexible clustering tool for summarization». In: (Feb. 2003) (cit. on p. 42).

[69] Regina Barzilay and Michael Elhadad. «Using Lexical Chains for Text Summarization». In: *Intelligent Scalable Text Summarization*. 1997. URL: https://www.aclweb.org/anthology/W97-0703 (cit. on p. 42).

[70] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. «Introduction to WordNet: An On-line Lexical Database*». In: *International Journal of Lexicography* 3.4 (Dec. 1990), pp. 235–244. ISSN: 0950-3846. DOI: 10.1093/ijl/3.4.235. eprint: https://academic.oup.com/ijl/article-pdf/3/4/235/9820417/235.pdf. URL: https://doi.org/10.1093/ijl/3.4.235 (cit. on p. 42).

[71] Som Gupta and S. K Gupta. «Abstractive summarization: An overview of the state of the art». In: *Expert Systems with Applications* 121 (2019), pp. 49–65. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2018.12.011. URL: https://www.sciencedirect.com/science/article/pii/S0957417418307735 (cit. on p. 43).

[72] Alexander M. Rush, Sumit Chopra, and Jason Weston. *A Neural Attention Model for Abstractive Sentence Summarization*. 2015. arXiv: 1509.00685 [cs.CL] (cit. on p. 43).

[73] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. «PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization». In: *CoRR* abs/1912.08777 (2019). arXiv: 1912.08777. URL: http://arxiv.org/abs/1912.08777 (cit. on p. 44).

[74] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. «Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer». In: *CoRR* abs/1910.10683 (2019). arXiv: 1910.10683. URL: http://arxiv.org/abs/1910.10683 (cit. on p. 44).

[75] Elena Baralis and Luca Cagliero. «Learning From Summaries: Supporting e-Learning Activities by Means of Document Summarization». In: *IEEE Transactions on Emerging Topics in Computing* 4.3 (2016), pp. 416–428. DOI: 10.1109/TETC.2015.2493338 (cit. on p. 45).

[76] Elena Baralis, Luca Cagliero, Alessandro Fiori, and Paolo Garza. «MWI-Sum: A Multilingual Summarizer Based on Frequent Weighted Itemsets». In: *ACM Trans. Inf. Syst.* 34.1 (Sept. 2015). ISSN: 1046-8188. DOI: 10.1145/2809786. URL: https://doi.org/10.1145/2809786 (cit. on p. 45).

[77] Guangbing Yang, Nian-Shing Chen, Kinshuk, Erkki Sutinen, Terry Anderson, and Dunwei Wen. «The effectiveness of automatic text summarization in mobile learning contexts». en. In: *Computers & Education* 68 (Oct. 2013), pp. 233–243. ISSN: 03601315. DOI: 10.1016/j.compedu.2013.05.012. URL: https://linkinghub.elsevier.com/retrieve/pii/S0360131513001334 (visited on 06/16/2021) (cit. on p. 45).

[78] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. *Pretrained Transformers Improve Out-of-Distribution Robustness*. 2020. arXiv: 2004.06100 [cs.CL] (cit. on p. 47).

[79] Sinno Jialin Pan and Qiang Yang. «A Survey on Transfer Learning». In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2010), pp. 1345–1359. DOI: 10.1109/TKDE.2009.191 (cit. on p. 47).

[80] S. Bozinovski. «Reminder of the First Paper on Transfer Learning in Neural Networks, 1976». In: *Informatica (Slovenia)* 44 (2020) (cit. on p. 47).

[81] Dani Yogatama et al. «Learning and Evaluating General Linguistic Intelligence». In: *ArXiv* abs/1901.11373 (2019) (cit. on pp. 47, 103).

[82] Alan Ramponi and Barbara Plank. «Neural Unsupervised Domain Adaptation in NLP—A Survey». In: *ArXiv* abs/2006.00632 (2020) (cit. on p. 48).

[83] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. *Zero-Shot Learning – A Comprehensive Evaluation of the Good, the Bad and the Ugly*. 2020. arXiv: 1707.00600 [cs.CV] (cit. on p. 49).

[84] Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. «Zero-Data Learning of New Tasks». In: *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*. AAAI'08. Chicago, Illinois: AAAI Press, 2008, pp. 646–651. ISBN: 9781577353683 (cit. on p. 49).

[85] Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. «A Hierarchical Network for Abstractive Meeting Summarization with Cross-Domain Pretraining». In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (Nov. 2020). URL: https://www.microsoft.com/en-us/research/publication/end-to-end-abstractive-summarization-for-meetings/ (cit. on pp. 51, 52, 69).

[86] Paco Nathan. *PyTextRank, a Python implementation of TextRank for phrase extraction and summarization of text documents*. 2016. DOI: 10.5281/zenodo.4602393. URL: https://github.com/DerwenAI/pytextrank (cit. on p. 55).

[87] Alexandra Savelieva, Bryan Au-Yeung, and Vasanth Ramani. «Abstractive Summarization of Spoken and Written Instructions with BERT». In: *CoRR* abs/2008.09676 (2020). arXiv: 2008.09676. URL: https://arxiv.org/abs/2008.09676 (cit. on p. 58).

[88] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. «Fine-Tuning Pretrained Language Models: Weight Initializations, Data Orders, and Early Stopping». In: *CoRR* abs/2002.06305 (2020). arXiv: 2002.06305. URL: https://arxiv.org/abs/2002.06305 (cit. on p. 58).

[89] Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. *Better Fine-Tuning by Reducing Representational Collapse*. 2020. arXiv: 2008.03156 [cs.LG] (cit. on p. 58).

[90] Potsawee Manakul and Mark Gales. *CUED - speech at TREC 2020 Podcast Summarisation Track*. 2021. arXiv: 2012.02535 [cs.CL] (cit. on p. 59).

[91] Siddhant Upasani, Noorul Amin, Sahil Damania, Ayush Jadhav, and A. Jagtap. «Automatic Summary Generation using TextRank based Extractive Text Summarization Technique». In: 2020 (cit. on p. 62).

[92] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. «Generating Long Sequences with Sparse Transformers». In: *CoRR* abs/1904.10509 (2019). arXiv: 1904.10509. URL: http://arxiv.org/abs/1904.10509 (cit. on p. 64).

[93] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. «Linformer: Self-Attention with Linear Complexity». In: *CoRR* abs/2006.04768 (2020). arXiv: 2006.04768. URL: https://arxiv.org/abs/2006.04768 (cit. on p. 64).

[94] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. «Reformer: The Efficient Transformer». In: *CoRR* abs/2001.04451 (2020). arXiv: 2001.04451. URL: https://arxiv.org/abs/2001.04451 (cit. on p. 64).

[95] Ilya Loshchilov and Frank Hutter. «Fixing Weight Decay Regularization in Adam». In: *CoRR* abs/1711.05101 (2017). arXiv: 1711.05101. URL: http://arxiv.org/abs/1711.05101 (cit. on pp. 68, 128, 129).

[96] Diederik Kingma and Jimmy Ba. «Adam: A Method for Stochastic Optimization». In: *International Conference on Learning Representations* (Dec. 2014) (cit. on pp. 68, 127, 128).

[97] Chin-Yew Lin. «ROUGE: A Package for Automatic Evaluation of summaries». In: Jan. 2004, p. 10 (cit. on p. 71).

[98] Feifan Liu and Yang Liu. «Exploring Correlation Between ROUGE and Human Evaluation on Meeting Summaries». In: *Audio, Speech, and Language Processing, IEEE Transactions on* 18 (Feb. 2010), pp. 187–196. DOI: 10.1109/TASL.2009.2025096 (cit. on p. 72).

[99] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. «BERTScore: Evaluating Text Generation with BERT». In: *CoRR* abs/1904.09675 (2019). arXiv: 1904.09675. URL: http://arxiv.org/abs/1904.09675 (cit. on p. 72).

[100] Adam Ek, Jean-Philippe Bernardy, and Stergios Chatzikyriakidis. «How does Punctuation Affect Neural Models in Natural Language Inference». In: *Proceedings of the Probability and Meaning Conference (PaM 2020)*. Gothenburg: Association for Computational Linguistics, June 2020, pp. 109–116. URL: https://www.aclweb.org/anthology/2020.pam-1.15 (cit. on p. 94).

[101] Eva García-Martín, Crefeda Faviola Rodrigues, Graham Riley, and Håkan Grahn. «Estimation of energy consumption in machine learning». In: *Journal of Parallel and Distributed Computing* 134 (2019), pp. 75–88. ISSN: 0743-7315. DOI: https://doi.org/10.1016/j.jpdc.2019.07.007. URL: https://www.sciencedirect.com/science/article/pii/S0743731518308773 (cit. on p. 114).

[102] Danny Hernandez Dario Amodei. «AI and Compute». In: (2018). URL: https://openai.com/blog/ai-and-compute/ (cit. on p. 114).

[103] Emma Strubell, Ananya Ganesh, and Andrew McCallum. *Energy and Policy Considerations for Deep Learning in NLP*. 2019. arXiv: 1906.02243 [cs.CL] (cit. on p. 114).

[104] Andy Jassy. «Amazon AWS ReInvent Keynote». In: (2018). URL: https://www.youtube.com/watch?v=ZOIkOnW640A (cit. on p. 114).

[105] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV] (cit. on pp. 126, 127).

[106] Olga Russakovsky et al. «ImageNet Large Scale Visual Recognition Challenge». In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y (cit. on p. 126).

[107] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. «Highway Networks». In: *CoRR* abs/1505.00387 (2015). arXiv: 1505.00387. URL: http://arxiv.org/abs/1505.00387 (cit. on p. 126).

[108] H. Robbins. «A Stochastic Approximation Method». In: *Annals of Mathematical Statistics* 22 (2007), pp. 400–407 (cit. on p. 127).