University of Manchester
School of Computer Science
Project Report 2023

Extractive Summarisation of UK Annual Reports

Author: Vladislav Yotkov

Supervisor: Dr. Jonathan Shapiro

Abstract

Extractive Summarisation of
UK Annual Reports

Author: Vladislav Yotkov

Although there has been considerable progress in Natural Language Processing (NLP) over the years, it has not fully reached the Accounting and Finance (AF) industry. In the meantime, companies worldwide produce vast amounts of textual data as part of their reporting packages to comply with regulations and inform shareholders of their financial performance. The glossy annual report is such an example, widely read by investors but it also tends to be quite long. Inspired by the Financial Narrative Summarisation (FNS) 2021 Task, we will design an Automatic Text Summarisation (ATS) system for the narrative parts of UK financial annual reports. With this goal in mind, we will implement and explore the following models for Extractive Text Summarisation (ETS): 1. custom Recurrent Neural Network (RNN), 2. fine-tuned FinBERT. In terms of evaluation, we will use the ROUGE metric to compare the performance of these models against standard ATS baselines: TextRank, and LexRank.

Supervisor: Dr. Jonathan Shapiro

Contents

List of Figures

List of Tables

0.1 Introduction

0.1.1 Financial Reports

Due to international regulations, companies are obliged to report their periodic performance (annual, bi-annual, quarterly) to various regulatory authorities[1] and other users (e.g., corporate stakeholders, investors, customers, suppliers, etc.). These reports contain essential information about the operations and finances of a business and are crucial for making informed decisions (from a user perspective), but are different in regulatory forms. For example,

1. 10-K reports filed to the SEC[2] and accessible through their Electronic Data Gathering, Analysis, and Retrieval[3] (EDGAR) system are only for US registered businesses. They follow a standardised template and are plain text, which makes them particularly easy for automated large-scale research ([EHAR^{+}19]). Also, the contents of these reports are strict, requiring solely five information sections[4].

2. UK annual reports, regulated by the UK's Financial Reporting Council (FRC), are typically the primary annual reporting method (also provided as PDF files). Unlike the 10-K, they are glossy and more stakeholder-oriented and enjoy unlimited discretion over non-mandated content ([EHAR^{+}19]) (e.g., photography and company brand material, non-mandatory narrative sections, etc.). However, these are more challenging for automated processing due to their variable section structure, formatting, and rich visual representations.

[1] Regulation authorities worldwide:

- Securities and Exchange Commission (SEC) in the USA
- European Securities and Markets Authority (ESMA) in Europe
- Financial Reporting Council (FRC) in the UK
- International Financial Reporting Standards (IFRS) in 167 jurisdictions worldwide

[2] https://www.sec.gov

[3] https://www.sec.gov/edgar

[4] (a) Business Overview (b) Risk Factors (c) Management's Discussion and Analysis of Financial Condition and Results of Operations (MD&A) (d) Financial Statements (e) Supplementary Disclosures

0.1.2 UK annual reports

The annual report is the primary corporate disclosure legally required for public companies by regulatory authorities. While it *does not have a rigid document structure* like the 10-K, it typically has a *narrative component*[5] and the financial statements (at the rear).

As we outlined in Section 0.1.1, UK annual reports have the following inconvenient properties with regard to large-scale text understanding.

- They are very long documents. Throughout the years, their average length has been increasing significantly with the number of pages rising 57% for the median report from 2003 to 2016 (47 to 74 pages, respectively) ([LY19]), due to additional regulations between 2006 and 2008 ([EHAR$^+$19].

- They have variable nomenclature. From firm to firm, naming conventions vary "dramatically", with more than 20 unique titles for various sections (e.g., Chair's letter to shareholders, Management Commentary) ([LY19]).

- They incorporate embedded info-graphics. While domain experts hail the integration of highly interactive elements into corporate reporting ([KB16]), the compilation to PDF makes the task of analysing such unstructured documents automatically even harder ([LY19]).

These challenges motivate the work of [EHRY$^+$19] who (a) established a set of 8 generic section headers[6] and (b) built the CFIE-FRSE[7] extraction tool that converts a text-based PDF annual report to simple text.

0.1.3 NLP in Accounting and Finance

The relevance of this project should also be understood from the perspective of the development of Natural Language Processing (NLP) in the Accounting and Finance (AF) domain.

[5]The narrative component of a UK annual report typically consists of 1. Management's Commentary 2. Letter to Shareholders 3. Corporate Governance Statement 4. Auditor's Report 5. Remuneration Report 6. Business Review 7. Environmental, Social, and Governance (ESG) Report 8. Risk Management Report

[6](a) Chairman Statement (b) CEO Review (c) Corporate Governance Report (d) Directors Remuneration Report (e) Business Review (f) Financial Review (g) Operating Review (h) Highlights

[7]The CFIE-FRSE stands for Corporate Financial Information Environment - Final Report Structure Extractor. It is publicly available at `https://github.com/drelhaj/CFIE-FRSE` and it can be used to convert English, Spanish and Portuguese annual reports.

As outlined in [Ell98], investors' trust in the accountability of businesses would be based no longer as much on just the financial statements, but also on more descriptive narratives that define strategy and planning of resource use. While some recognise the importance of understanding in-domain textual information ([L$^+$10]), others like [EHRW$^+$19] report that the industry is still doubtful and cynical about the NLP applications in the analysis of financial market disclosures. Furthermore, the latter also observe that AF researchers rely extensively on bag-of-words models, which are *not sufficient to encode complex contextual and semantic meaning* (especially in a domain with such *specialized language*). As for ATS [CHW19] is said to be the single AF study into disclosure summarisation. It demonstrates that machine-generated summaries are less likely to bias positively investor decisions compared to managerial ones. Therefore, this only confirms the existence of a wide gap in NLP applications in Accounting research, which further motivates our work.

0.1.4 Financial Narrative Summarisation 2021 (FNS21) Task

The FNS Task is part of the annual Financial Narrative Processing (FNP) Workshop [8] organised by Lancaster University since 2018, which aims to:

- encourage the advancement of financial text mining & narrative processing
- examine methods of structured content retrieval from financial reports
- explore causes and consequences of corporate disclosure

as stated in their inaugural proceedings [9].

For that purpose, they produce datasets of extracted narratives (with the help of the CFIE-FRSE tool) from annual reports of UK companies listed on the London Stock Exchange (LSE).

In their FNS21 Task, there were $3,863$ such reports (Table 1), while the average length was reported at 80 pages, and the maximum of more than 250 pages ([LV21]).

Additionally, for every report, there were at least two gold summaries situated in the

[8] https://wp.lancs.ac.uk/cfie/
[9] https://wp.lancs.ac.uk/cfie/fnp2018/

annual report itself [10] The workshop's goal was to build ATS systems that generate a single summary for an annual report, no longer than $1,000$ words (almost just as long as the gold summaries on average).

Data Type	Training	Validation	Testing	Total
Report full text	3,000	363	500	3,863
Gold summaries	9,873	1,250	1,673	12,796

Table 1: FNS21 Data Split

We acknowledge that due to the scarcity of publicly available financial data this third-year project could not have been possible without the kind permission of the FNP organisers to use the training and validation datasets from their FNS21 Task ([EHRZ21]).

[10] The gold summaries being already in the annual report is not problematic because these reports are already written by domain experts who know how to summarise the financial state of a company. Hence, multiple sections/paragraphs could achieve this thoroughly, and the organisers have identified & extracted them manually with the help of the professional writers of the individual reports. At this moment, one can begin to doubt the point of applying ATS techniques, but due to the *lack of rigid document structure*, *it is not trivial to automatically find these text excerpts with heuristic methods*. Furthermore, we can formulate this challenge as finding the latent features of a summarising (i.e., "to-be-in-the-summary") sentence, highlighted as one of the fundamental advantages of NLP in AF research ([LY19], [EHRW$^+$19]).

0.2 Background

0.2.1 Supervised Learning

0.2.2 TFIDF

0.2.3 Word Embeddings

Historically, to represent a token (i.e., word) w_i in a vocabulary V numerically, we define a one-hot-encoding vector of all zeroes except of a one at the index of the word w_i in V (i.e., i).

The results are sparse individual word vectors being orthogonal to each other which 1. waste memory (each word is a $|V|$-sized vector, hence a total of $|V|^2$ for all tokens) and more importantly 2. fail to encode semantic similarity due to their cosine similarity being always zero.

Traditionally, AF research has represented an input text with the help of bag-of-words (BOW) models which can be viewed from the 1. the binary perspective - represent a whole document d as a binary vector containing ones for all words w_i occurring in d from V, 2. the term frequency perspective - encode number of word occurrences in documents instead of binary representation ([XCWS13]), and 3. the tf-idf perspective - extend the latter to penalise ubiquitous terms ([Jon72]). Nevertheless, these vectors are very sparse and unable to encode more complex contextual and semantic meaning.

To address these shortcomings *short*[11] and *dense*[12] word embeddings like Word2Vec ([MCCD13]) and FastText ([BGJM17]) have been developed. In [MCCD13] the authors manage to condense the vector space and ensure that word representations have *multiple degrees of similarity* ([MYZ13]) (e.g., semantic - the meaning of words, morphological - structure of sub-words, etc).

Furthermore, the proposed models - CBOW (Continuous Bag of Words[13]) and Skip-gram evidently capture subtle semantic relationships and allow intuitive arithmetic opera-

[11]i.e., with a small number of dimensions

[12]i.e., continuous real-numbered values instead of 0/1s

[13]CBOW naming is derived from 1. the continuous distributed representation of the context and 2. the projection layer being shared across context words, i.e., the order of words does not affect the projection (similar to how bag-of-words model fails to encode word order).

tions as shown in the popular analogy: $\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}} \approx \overrightarrow{\text{queen}}$[14].

The intuition for the two Word2Vec models is that in CBOW, the context (i.e., the surrounding tokens) is used to predict the middle token, while in skip-gram, the input token is used to predict the context (i.e., the surrounding tokens) (Figure 1).

Meanwhile, internally, the context prediction is cast as a binary classification task with positive examples being the target word and its surroundings, whereas the negatives ones are generated through random sampling from the dictionary. Then, the CBOW embeddings are the learned weights of a logistic regression classifier with future and history words (i.e., the context window) as the input and the goal of correctly classifying the word in-between. In contrast, the Skip-gram uses the middle word as an input to the classifier and predicts the individual context words around it.

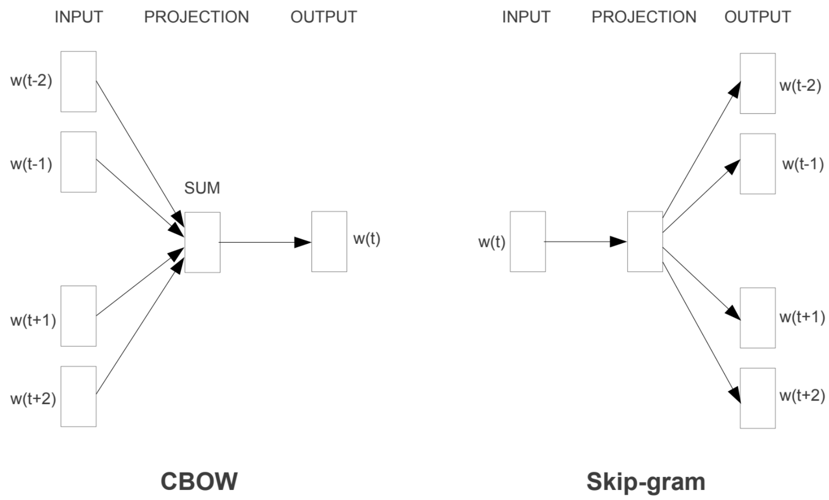


Figure 1: Word-from-context and context-from-word prediction in CBOW and Skip-gram, respectively. Figure is from [MCCD13]

.

A downside to the Word2Vec models is that they cannot handle out-of-vocabulary (OOV) word tokens, i.e., they cannot generate an embedding vector for words missing from the training data, which is crucial in real-life problems with *noisy* input or morphologically rich languages. For this reason [BGJM17], propose FastText as an extension to the Skip-gram model that makes use of character-level information to deal with unknown

[14]For a formal explanation on how analogies are realised in word embeddings we direct the readers to [EDH19]

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

Figure 2: Attention calculation (Query, Key, Value)

tokens. Here, each word is itself a bag of character n-grams which captures meaning of prefixes, suffixes, and morphemes. Additionally, two symbols are further introduced to mark the beginning and the end of a token, and help differentiate between sub-words and short words. For example, the character trigrams of the word *believe* are *<be, bel, eli, lie, iev, eve, ve>* where the sub-word *lie* is different from the word token *<lie>*.

Therefore, the final target word embedding is the sum of its constituent character n-grams which are learned via the Skip-gram model. This makes FastText very convenient for representing unknown words as the sum of *static* constituent n-grams ([JM00]).

Nevertheless, when it comes to domain-specific problems, general pre-trained word embeddings do not perform very well [RZP21]. demonstrate that even state-of-the-art embedding models like Google's Word2Vec(skip-gram)[15] and Facebook's FastText(skip-gram)[16]trained on 100 billion and 16 billion words, respectively, struggle to understand financial language like 1. *apple* standing for the company *Apple*, 2. ticker analogies, e.g., *amazon* is to *X* as *microsoft* is to *msft*, 3. grouping company name to ticker, exchange and country. In the same paper, the authors propose using the same algorithms (i.e., CBOW, Skip-gram, and FastText) but training solely on financial data instead - 15 years of financial news from the Dow Jones Newswires Text News Feed database, to produce the FinText models. They report a substantial increase in performance in and sensitivity to detecting financial jargon and relationships. In our project we acknowledge that a purpose-built financial word embedding (trained on proprietary data) will be more beneficial and more suited for the task of text summarisation of annual financial reports, which is why we select FinText as our preferred model.

0.2.4 Attention

0.2.5 Recurrent Neural Networks (RNNs)

The vanilla RNN is a basic type of RNN architecture designed for processing sequential data. It learns temporal patterns from the initial data by looping over the hidden layers which allow information to persist (i.e., they serve as a network memory) ([Ola15]). The key component is the recurrent hidden state h_i (Figure 3a) updated at each time step using input data and the previous hidden state (Eq.2). This allows the RNN to capture contextual information and temporal dependencies in the sequence. However, due to the inherent vanishing and exploding gradient problems with the vanilla RNNs, they have limited ability to learn long-term dependencies ([BSF94]). To resolve these issues more advanced RNN architectures like LSTMs and GRUs have been developed.

$$h_t = f(W_{hh}h_{t-1} + W_{xh}x_t + b_h)^{17} \tag{2}$$

$$y_t = g(W_{hy}h_t + b_y) \tag{3}$$

The Long Short-Term Memory (LSTM) recurrent neural network has become a ubiquitous method in sequential problems (e.g., language modelling, time series forecasting). This is so because it allows long-term dependencies to propagate through the network with the help of control gates *input* and *forget*, which reduce the effect of the vanishing gradient issue in the vanilla RNN[18].

A more simple variant of the LSTM is the Gated Recurrent Unit (GRU) which combines the *input* and *forget* gates into an *update* gate for a model with fewer parameters and faster training ([CCG21]). Nevertheless, due to the sequential nature of the LSTM, the training

[15] `https://code.google.com/archive/p/word2vec/`

[16] `https://fasttext.cc/`

[17] The algebraic formulation of the vanilla RNN has the following variables: 1. x_t is the input at time step t. 2. h_{t-1} is the hidden state at time step $t-1$. 3. h_t is the hidden state at time step t. 4. y_t is the output at time step t. 5. f and g are activation functions for the hidden and output layers, respectively. 6. W_{hh}, W_{xh}, W_{hy} are weight matrices for the hidden-to-hidden, input-to-hidden, and hidden-to-output connections, respectively. 7. b_h and b_y are bias terms for the hidden and output layers, respectively.

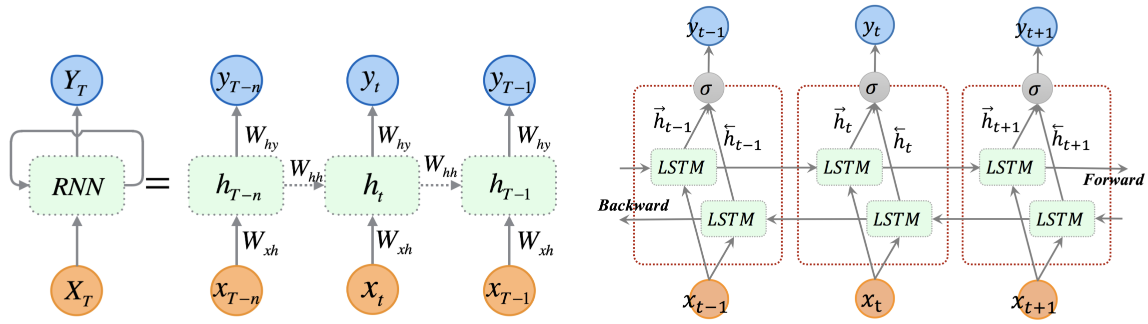
[18] We direct readers to [Bay15] where the authors demonstrate that the LSTM's "temporal" gradient is unaffected by the fixed weight factor W of the vanilla RNN that is driving the derivative to zero. This is ensured by the additional architecture unit the *forget* gate, which learns to control the gradient flow in the network.

process cannot be parallelised across GPUs, i.e., the learning cannot be made quicker by more computational resources.

As noted by [GMH13], a limitation of the unidirectional RNNs is that they only make use of previous context in the sequence. To alleviate this and establish more complex relationships between words [SP97], propose a bi-directional architecture consisting of a forward and a backward RNN.

We can therefore represent the hidden layers (h_i) per time-step (where T is the sequence size) with the following notation: 1. $(\overrightarrow{h_1}, \ldots, \overrightarrow{h_T})$ are the forward hidden states from left to right (i.e., $x_1, \ldots, x_T$) and 2. $(\overleftarrow{h_1}, \ldots, \overleftarrow{h_T})$ are the backward hidden states from right to left (i.e., $x_T, \ldots, x_1$) (Figure 3b).

Then, for a single word x_i, its respective *annotation* (i.e., condensed representation) is constructed by the concatenation of the forward and backward hidden states - $h_i = [\overrightarrow{h_i^T}; \overleftarrow{h_i^T}]^T$ as specified in [BCB16].



(a) Unfolded RNN

(b) Unfolded bi-directional LSTM

Figure 3: Unfolded recurrent architectures ([CKW18])

0.2.6 Encoder-Decoder and Attention

However, to deal with many-to-many sequence-to-sequence problems (e.g., machine translation, speech recognition, abstractive text summarisation) a new type of neural architecture is necessary [SVL14]. and [CvMG+14] introduced the Encoder-Decoder network (Figure 4) with 1. the encoder being an RNN that maps an input sequence $(x_1, x_2, \ldots, x_n)$ to a continuous fixed-length context vector c and 2. the decoder, also an RNN, taking this vector and producing an output sequence $(y_1, y_2, \ldots, y_m)$. They train the two RNNs jointly, maximising the conditional probability of the target sequence given a source sequence, i.e., $p(y_1,\ldots,y_{T'}|x_1,\ldots,x_T)$. For the selection of the neural model the authors had naturally chosen LSTM and GRU, respectively, due to the resolution of the vanishing gradient problem (as discussed in Section 0.2.5) and for the time being [SVL14] managed to achieve state-of-the-art results in machine translation.

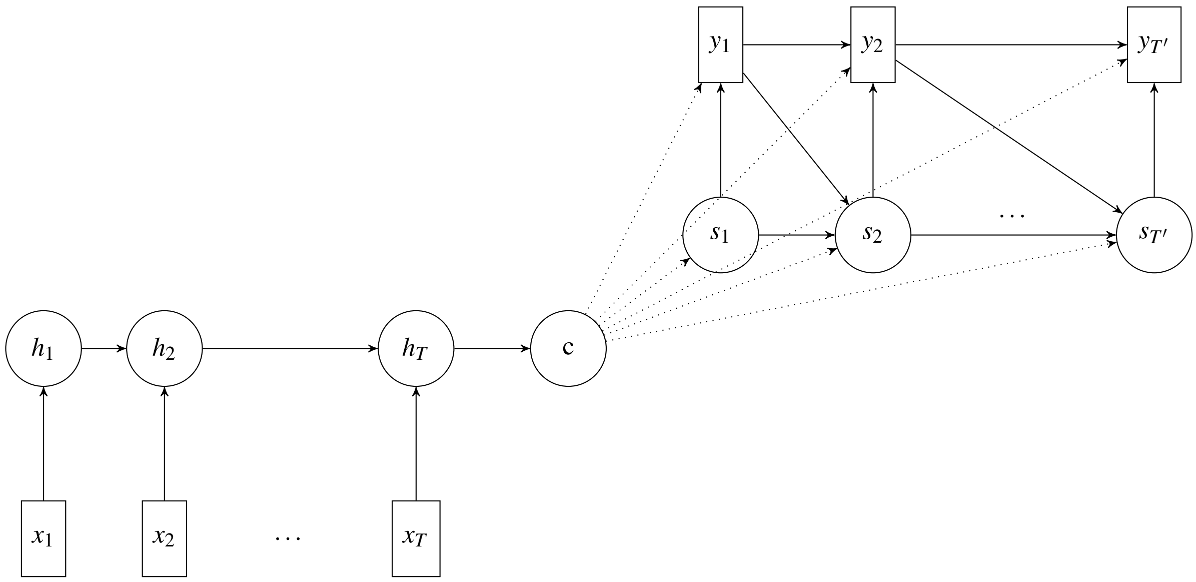


Figure 4: Encoder-Decoder Schema

It is in [BCB16] where the authors suppose that the fixed-length vector results in a bottleneck such that the longer the sequences, the worse the compression performance is for neural networks. Meanwhile, they propose to replace the fixed-length vector with a variable-length one which searches for the most relevant information from the source sequence. This is achieved through an alignment model which accepts as input the produced annotations from the encoder $h_1,\ldots,h_T$ (see Section 0.2.5 for details on how these are generated). Then the context vector c_i which is now distinct for each input word (unlike in [CvMG$^+$14, SVL14]) becomes a weighted sum[19] of the annotations h_j (Eq.4) and each weight α_{ij} is calculated as the normalized attention score (Eq.5). The alignment model is trained jointly with the encoder-decoder system and intuitively it evaluates the importance

[19] And hence it is known as the *additive attention*.

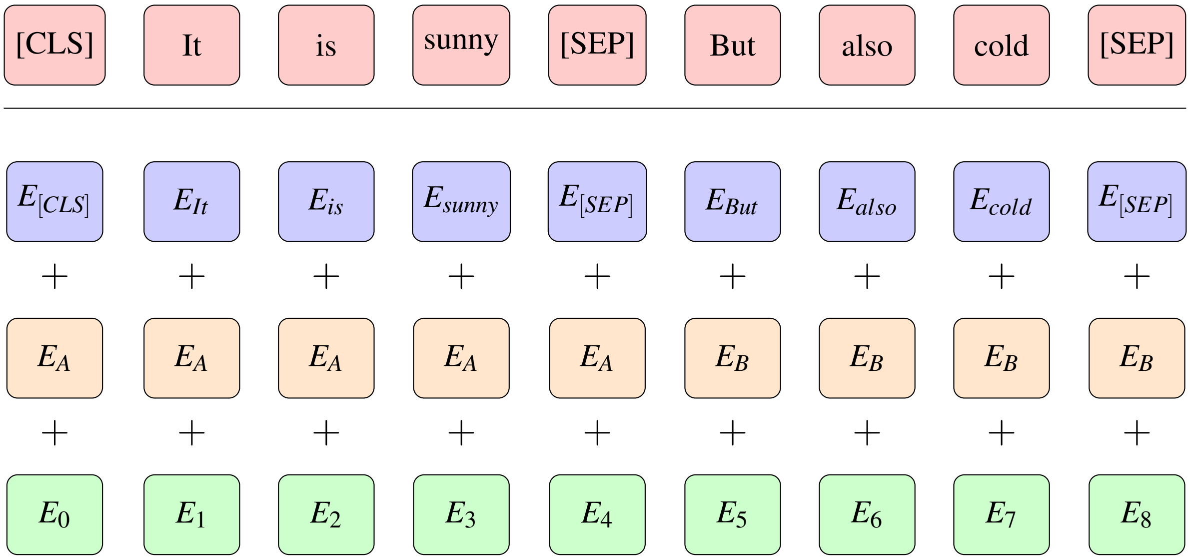


Figure 5: BERT: Input Embeddings

e_{ij} of an annotation h_j in generating a new state s_i and output token y_i (Eq.6).

$$c_i = \sum_{j}^{T} \alpha_{ij} h_j \tag{4}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T} \exp(e_{ik})} \tag{5}$$

$$e_{ij} = a(s_{i-1}, h_j) \tag{6}$$

0.2.7 Transformers

The Transformer ([VSP+17]) is another sequence-to-sequence architecture which is parallelisable and attention-based.

0.2.8 Text Summarisation

Text summarisation is the task of transforming a piece of text into a shorter version that retains the most important information. There are two overarching categories: extractive and abstractive text summarisation. The former formulates the problem as a subset selection problem by returning only the most salient text excerpts from the original document ([ZLC$^+$20]), while the latter aims to generate content anew, similar to how humans would do.

We will outline some key models that inspired our work below:

- **Gokhan**: The authors employ an unsupervised summariser based on K-Means clustering of sentences encoded with SentenceBERT ([RG19]). However, their embeddings are pre-trained on general text, and they suggest that employing in-domain language models would result in a better performance.

- **AMUSE** ([LV21]): The authors design an ETS system comprised of the following steps 1. shortening of report with an existing Genetic Algorithm [LL13], 2. encoding sentences with BERT vectors, and 3. performing binary classification with LSTMs for salient sentence extraction. They suggest that further work should incorporate 1. efficient preliminary sentence removal, and 2. additional neural modelling stages for the representation and detection of relevant input text parts.

- **Hybrid model with RL** ([ZSEHR21]): The authors train a joint extractive-abstractive summarisation model with reinforcement learning optimised for the ROUGE-2 F1 metric. Their networks are based on attentive LSTMs augmented with an additional copy mechanism ([VFJ15]) achieving the second highest F1 score in the FNS21 competition.

- **T5** Hybrid ([Orz21]): The author used T5 ([RSR$^+$20]) for a hybrid model fine-tuned to generate the beginning of an abstractive summary and find the closest match of the output in the report's full text. This is the best performing algorithm in the FNS21 competition but also the first to consider transformer models from an abstractive summarisation perspective in the FNP workshops so far.

In this work we will be solely exploring the extractive method, and more specifically - the *supervised neural-based* (i.e., RNN, Transformer) type and the *unsupervised graph-based* (i.e., TextRank, LexRank) type.

0.2.9 LexRank

LexRank ([ER04]) is an unsupervised extractive summarisation method consistently used as a baseline in the FNS21 and previous challenges. It retrieves the most salient document sentences by computing their importance based on *eigenvector centrality*. To do that the algorithm creates a graph where each sentence represents a node and each edge is a weight between two nodes ([SGWM20]). The sentences are encoded as bag-of-words vectors of size N - the vocabulary size, and the weight metric is a combination of tf-idf (Eq.7,8) and cosine similarity - Eq.9.

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{7}$$

$$\text{tf-idf}(t,d,D) = \mathrm{tf}(t,d) \cdot \mathrm{idf}(t,D) \tag{8}$$

$$\text{tf\_idf\_cosine\_similarity}(s_1,s_2) = \frac{\sum_{t \in T} \text{tf-idf}(t,s_1,D) \cdot \text{tf-idf}(t,s_2,D)}{\sqrt{\sum_{t \in T} \text{tf-idf}(t,s_1,D)^2} \cdot \sqrt{\sum_{t \in T} \text{tf-idf}(t,s_2,D)^2}} \tag{9}$$

where t is a term, d is a document within a collection of documents/sentences D.

Also, s_1 and s_2 are two sentences and T represents the set of all terms in both of them while $tf(t,d)$ denotes the term frequency of t in d, and $idf(t,D)$ is the inverse document frequency of t in the collection D.

The authors further propose finding the most important sentences by 1. applying a threshold for the creation of edges with Eq.9, 2. building an adjacency matrix and normalizing it to produce *transition probabilities*, 3. computing in an iterative fashion the *eigenvector centrality* until convergence, and finally 4. ranking sentences based on their *lexical* PageRank ([BP98]) score.

0.3 Design & Development

0.3.1 Methodology

We approach the annual report summarisation problem from a supervised perspective - we cast the task of Extractive Text Summarisation (ETS) as a binary classification problem defined on the sentence level. More formally, we can describe the annual report as $d = \{s_1, s_2, \ldots, s_n\}$, where d is a document, represented in terms of sentences $s_i,\ 1 \leq i \leq n$ ([Liu19]).

Then, a candidate summary[20] can be $c = \{s_1, s_2, \ldots, s_k | s_i \in d\},\ 0 \leq k \leq n$.

We further need to define the *gold summary*, c^* for a document d.

In the case of the FNS21 task, there are at least two summaries per report, hence we will use the following notation for the set of all gold summaries for each document $C^* = \{c_1^*, c_2^*, \ldots, c_p^*\}$. Furthermore, the supervised learning labels are $y_i \in \{1, 0\}$ for each sentence s_i in d if the sentence is or is not in ***any***[21] of the gold summaries c_j^* for that document.

In general, to assess the quality of a candidate summary c, we measure its similarity with the gold summary c^* based on their n-gram overlap $R = (c, c^*)$, where R is the ROUGE-F_1[22] metric([Lin04]).

For the FNS21 task due to the extractive nature of our approach we will evaluate our models based on the ROUGE-maximising c_i^* gold summary[23], i.e.,

While some authors ([ZSEHR21]) follow the greedy ROUGE-maximisation method

[20] A candidate summary is generated from a model m_i but it is not yet a *best summary*.

[21] To increase the positive samples (i.e., the summarizing sentences) we do not restrict ourselves to just one gold summary in the training process unlike [Orz21]. Our goal is to achieve better latent feature extraction of summaries through the employment of all existing data. However, we are aware that this approach is more likely to encounter standard ETS issues, specifically - extracted summary sentences could be retrieved from unrelated paragraphs in the report. This causes the "dangling anaphora" phenomenon, i.e. decontextualised extracts are stitched together and could mislead the reader due to out-of-context references as specified in [Lin09].

[22] We use a slightly different but faster version of ROUGE compared to the official metric [Lin04]. It can be accessed at: `https://github.com/pltrdy/rouge`

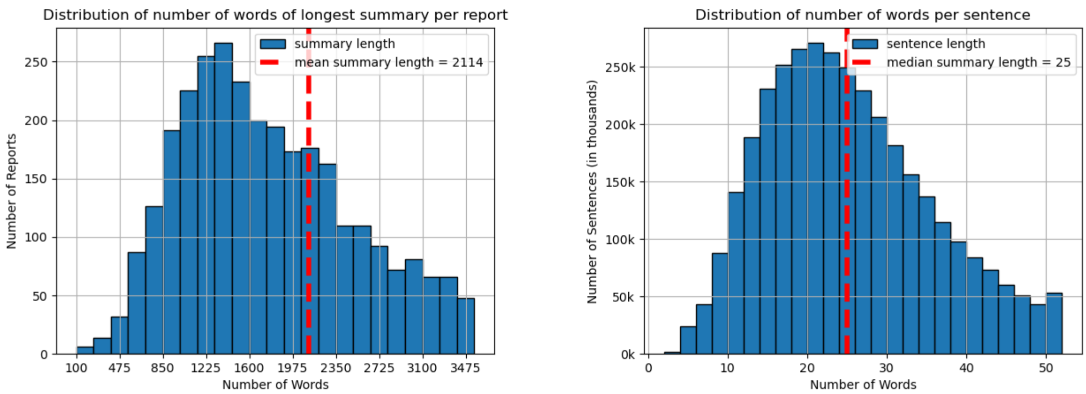
[23] The intuition is that by extracting multiple sentences from the report, our generated candidate summary can retain sentences from *any* of the gold summaries. Hence, there must be at least one such gold summary where the overlap is maximal. The practical implications are that two models, m_1 and m_2 can produce two different candidate summaries c_1 and c_2, respectively. Their individual evaluation is based on gold summaries c_1^* and c_2^* (which can be the same when the candidates c_1 and c_2 are identical).

$$r = \underset{c^* \in C^*}{\operatorname{argmax}} R(c, c_i^*) \tag{10}$$

Figure 6: Candidate summary evaluation as a gold summary ROUGE-maximisation

of matching summary sentences to document sentences (established in [NZZ17]), we approach the problem in a more practical and faster fashion. After manual observation of the reports against their gold summaries, it became clear that for almost all sentences of c_i^*, there was an exact match with a sentence in the whole annual report d.

This hypothesis was proven correct by one of the FNS21 contestants ([Orz21]) who reported that 99.4% of the summaries were included in the report as whole subsequences. Hence, after having pre-processed the text documents we iteratively match the sentences and generate the binary classification labels ($\{1,0\}$ representing *summary* and *non-summary*, respectively) for both the training and testing datasets.



(a) Number of words in longest report summary (b) Number of words in training sentences

Figure 7: Distribution of number of words in training sentences and report summaries

0.4 Evaluation

0.4.1 Confusion Matrix

The confusion matrix is an essential tool to visualise and help assessing the performance of trained classifiers against the true labels y_i.

For the problem of binary classification, it is a square matrix (Table 2) that displays the following key elements:

- True Positives (TP): Correct predictions of the positive class.
- True Negatives (TN): Correct predictions of the negative class.
- False Positives (FP): Incorrect predictions of the positive class (Type I error).
- False Negatives (FN): Incorrect predictions of the negative class (Type II error).

where for the problem of extractive text summarisation, the positive and negative classes correspond to *summary* and *non-summary* sentences, respectively. These matrix elements can be further combined into informative classification metrics:

- Accuracy: Proportion of correctly classified instances out of the total instances. Formulated as $\frac{TP+TN}{TP+TN+FP+FN}$.
- Precision (or Positive Predictive Value): Proportion of true positive instances out of all instances predicted as positive. Formulated as $\frac{TP}{TP+FP}$.
- Recall (or Sensitivity): Proportion of true positive instances out of all actual positive instances. Formulated as $\frac{TP}{TP+FN}$.
- F1-score: Harmonic mean of precision and recall (i.e., the trade-off between the two). Formulated as $2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$.

	Actual	
	Positive	**Negative**
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Table 2: Confusion Matrix

$$ROUGE-N = \frac{\sum_{S\in R}\sum_{n-gram\in S} count_{match}(n-gram)}{\sum_{S\in R}\sum_{n-gram\in S} count(n-gram)} \tag{11}$$

Figure 8: ROUGE-N: N-gram Co-Occurrence Statistics

Bibliography

[Bay15] Justin Simon Bayer. Learning sequence representations, 2015.

[BCB16] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[BGJM17] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30:107–117, 1998.

[BSF94] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[CCG21] Roberto Cahuantzi, Xinye Chen, and Stefan Güttel. A comparison of LSTM and GRU networks for learning symbolic sequences. *CoRR*, abs/2107.02248, 2021.

[CHW19] Eddy Cardinaels, Stephan Hollander, and Brian White. Automatic summarization of earnings releases: attributes and effects on investors' judgments. *Review of Accounting Studies*, 24, 09 2019.

[CKW18] Zhiyong Cui, Ruimin Ke, and Yinhai Wang. Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction. *CoRR*, abs/1801.02143, 2018.

[CvMG+14] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics.

[EDH19] Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. Towards understanding linear word analogies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3253–3262, Florence, Italy, July 2019. Association for Computational Linguistics.

[EHAR+19] Mahmoud El-Haj, Paulo Alves, Paul Rayson, Martin Walker, and Steven Young. Retrieving, classifying and analysing narrative commentary in unstructured (glossy) annual reports published as pdf files. *Accounting and Business Research*, 2019. Forthcoming.

[EHRW+19] Mahmoud El-Haj, Paul Rayson, Martin Walker, Steven Young, and Vasiliki Simaki. In search of meaning: Lessons, resources and next steps for computational analysis of financial discourse. *Journal of Business Finance & Accounting*, 46(3-4):265–306, 2019.

[EHRY+19] Mahmoud El Haj, Paul Edward Rayson, Steven Eric Young, Paulo Alves, and Carlos Herrero Zorita. *Multilingual Financial Narrative Processing: Analysing Annual Reports in English, Spanish and Portuguese*. World Scientific Publishing, 2019.

[EHRZ21] Mahmoud El-Haj, Paul Rayson, and Nadhem Zmandar, editors. *Proceedings of the 3rd Financial Narrative Processing Workshop*, Lancaster, United Kingdom, 15-16 September 2021. Association for Computational Linguistics.

[Ell98] Robert K Elliott. Accounting in the 21st century. 1998.

[ER04] Gunes Erkan and Dragomir R. Radev. Lexrank: Graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 2004.

[GMH13] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013.

[JM00] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, USA, 1st edition, 2000.

[Jon72] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

[KB16] P Kriz and H Blomme. The future of corporate reporting–creating the dynamics for change. *International Federation of Accountants (IFAC), available at: www. ifac. org/global-knowledgegateway/viewpoints/future-corporate-reporting-creating-dynamics-change (accessed 29 May 2016)*, 2016.

[L+10] Feng Li et al. Textual analysis of corporate disclosures: A survey of the literature. *Journal of accounting literature*, 29(1):143–165, 2010.

[Lin04] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[Lin09] Jimmy Lin. Summarization. In *Encyclopedia of Database Systems*, pages 2906–2910. Springer, Heidelberg, Germany, 2009.

[Liu19] Yang Liu. Fine-tune bert for extractive summarization. *arXiv preprint arXiv:1903.10318*, 2019.

[LL13] Marina Litvak and Mark Last. Multilingual single-document summarization with MUSE. In *Proceedings of the MultiLing 2013 Workshop on Multilingual Multi-document Summarization*, pages 77–81, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[LV21] Marina Litvak and Natalia Vanetik. Summarization of financial reports with AMUSE. In *Proceedings of the 3rd Financial Narrative Processing Workshop*, pages 31–36, Lancaster, United Kingdom, 15-16 September 2021. Association for Computational Linguistics.

[LY19] Craig Lewis and Steven Young. Fad or future? automated analysis of financial text and its implications for corporate reporting. *Accounting and Business Research*, 49(5):587–615, 2019.

[MCCD13] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[MYZ13] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[NZZ17] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[Ola15] Christopher Olah. Understanding lstm networks, 2015.

[Orz21] Mikhail Orzhenovskii. T5-LONG-EXTRACT at FNS-2021 shared task. In *Proceedings of the 3rd Financial Narrative Processing Workshop*, pages 67–69, Lancaster, United Kingdom, 15-16 September 2021. Association for Computational Linguistics.

[RG19] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[RSR+20] 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. 21(1), 2020.

[RZP21] Eghbal Rahimikia, Stefan Zohren, and Ser-Huang Poon. Realised volatility forecasting: Machine learning via financial word embedding. *SSRN Electronic Journal*, preprint:1–20, July 2021.

[SGWM20] Steven Shearing, Abigail S. Gertner, Ben Wellner, and Liz Merkhofer. Automated text summarization: A review and recommendations. 2020.

[SP97] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks, 2014.

[VFJ15] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[XCWS13] Zhixiang Eddie Xu, Minmin Chen, Kilian Q. Weinberger, and Fei Sha. An alternative text representation to tf-idf and bag-of-words. *ArXiv*, abs/1301.6770, 2013.

[ZLC+20] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online, July 2020. Association for Computational Linguistics.

[ZSEHR21] Nadhem Zmandar, Abhishek Singh, Mahmoud El-Haj, and Paul Rayson. Joint abstractive and extractive method for long financial document summarization. In *Proceedings of the 3rd Financial Narrative Processing Work-*

shop, pages 99–105, Lancaster, United Kingdom, 15-16 September 2021. Association for Computational Linguistics.