Combining financial word embeddings and knowledge-based features for financial text summarization: UC3M-MC System at FNS-2020

Jaime Baldeón[1], Paloma Martínez[1] and Jose L. Martínez[1, 2]
[1]Computer Science Department,
Universidad Carlos III de Madrid, Spain
[2]MeaningCloud LLC, USA
100332810@alumnos.uc3m.es
pmf@inf.uc3m.es
jmartinez@meaningcloud.com

Abstract

This paper describes the systems proposed by HULAT research group from Universidad Carlos III de Madrid (UC3M) and MeaningCloud (MC) company to solve the FNS 2020 Shared Task on summarizing financial reports. We present a narrative extractive approach that implements a statistical model comprised of different features that measure the relevance of the sentences using a combination of statistical and machine learning methods. The key to the model's performance is its accurate representation of the text, since the word embeddings used by the model have been trained with the summaries of the training dataset and therefore capture the most salient information from the reports. The systems' code can be found at https://github.com/jaimebaldeon/FNS-2020.

1. Introduction

The remarkable extension and diversity of textual and numerical structures in financial reports hinder the summarization process. Therefore, it is paramount to filter the relevant information and represent it accurately to produce high quality summaries. With this objective, we have developed a narrative extractive model with gold standard summaries knowledge.

What makes the proposed approach interesting is its outstanding capacity to capture the relevant information from the gold standard summaries and extract (according to some parameters) the most salient sentences from the reports.

It is the first time the team (HULAT) has participated in the FNS task and our goal was to explore features based on word embeddings that could understand and represent financial reports with higher precision. Therefore, the system implements a combination of statistical and machine learning techniques to determine the salience of the sentences in order to extract the most relevant ones from the original financial report.

2. Related work

Automatic text summarization can be tackled with two different approaches: extractive and abstractive summarization. The former approach is the most commonly used, since it is less computationally expensive and not as complex as abstractive summarization, furthermore, its performance has shown great results and settled the base to further research and development. However, it still faces some drawbacks, such as lack of cohesion between sentences or balance, and its methodology is still not close enough to human-like text processing, which is more similar to an abstractive approach.

Extractive summarization has implemented multiple different approaches: statistical approaches as purely statistical methods, statistical and linguistic features, and structure representation methods; and machine learning approaches such as Naïve Bayes, Decision Trees, Neural Networks, etc. (Spärck,

Barcelona, Spain (Online), December 12, 2020.

2007). As for abstractive summarization, state of the art approaches have been carried out bringing automatic summarization to the cutting edge: Recurrent Neural Networks, Long Short-Term Memory networks (Hochreiter and Schmidhuber, 1997), Sequence to sequence models (Nallapati et al., 2016) and Transformers (Vaswani et al., 2017). From the point of view of evaluation (Ermakova et al, 2019) provides an overview of different international challenges covering summarization tasks in several domains such as DUC (Document Understanding Conference), TAC (Text Analysis Conference) and CLEF (Cross Language Evaluation Forum). A detailed review of evaluation measures (informativeness and readability evaluations) is also given, as well as an automatic framework for evaluating metrics that does not require human annotations to face the problem of laborious and time-consuming generation of annotated corpora.

Depending on the source of the documents, (Xu et al, 2013) and (Li and Zhang, 2020) introduce approaches for tweet summarization, (Cagliero and La Quatra, 2020) describes an extractive supervised approach for scientific articles summarization and (Fabbri et al, 2019) introduces an abstractive approach for news summarization. In the financial domain, however, previous work is almost non-existent; participants in FinTOC-2019 shared task, (Juge et al. 2019), developed systems for title and TOC (table of content) extraction, multi-class text classification and tone detection but not for report summaries extraction. Further research in financial narrative processing with the capacity of analyzing large reports is required, hence the need to explore new approaches to tackle this challenge.

3. Architecture

Aiming to solve the FNS 2020 task, two systems have been developed by the team: HULAT-1 and HULAT-2. The solution proposed is described in Figure 1. Both systems follow the same approach: a statistical model based on different features that measure the relevance of the sentences in different aspects. The main component of the systems is the Summarizer (see Methods section for a detailed description of the component). The difference between HULAT-1 and HULAT-2 lies within this component (Figure 1), where the first system employs sentence position, keyword occurrences and gold standard similarity as features for the model, whereas the second one uses topic extraction instead of keyword occurrences.

4. Methods

The HULAT systems follow a three-stage pipeline: text processing, feature extraction and modelling (Spärck, 2007). However, each system has been divided into different main components: Fin2Vec, Gold Standard Vectorizer and Summarizer.

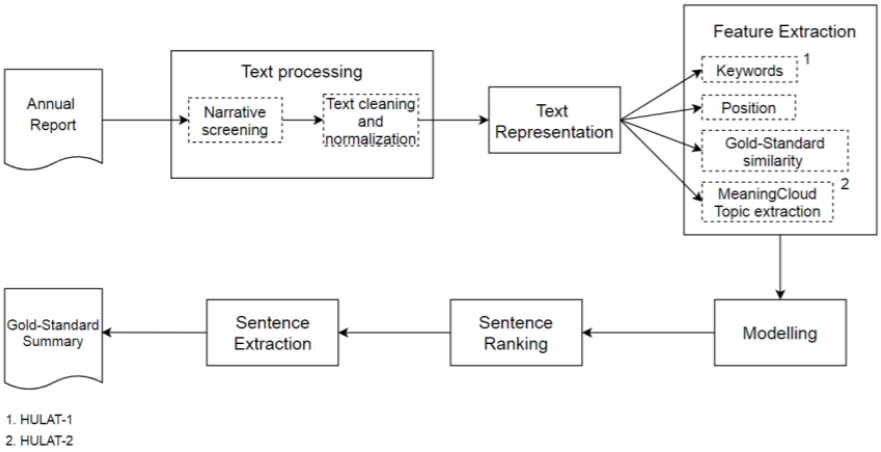


Figure 1. HULAT-1 and HULAT-2 Summarizers overview.

5. HULAT-1

The approach used is a combination of statistical and machine learning methods, since word embeddings generated with a machine learning model are used to produce one of the features that are fed into the statistical model. This approach enables the system to capture insights from the text based on semantic knowledge, which is provided by the representation of the document —using word embeddings— that captures the cohesion and coherence between sentences. Furthermore, the system is able to accurately understand the text since the embeddings have been trained with the specific financial vocabulary of the training dataset summaries. Therefore, sentences are evaluated based on their similarity to this training dataset knowledge captured by the embeddings, thus increasing the precision of the model.

The **Fin2Vec component** analyses financial text and represents it numerically with vectors using a neural network Word2Vec model. Therefore, the embedding model, fed with words from the training dataset summaries, has been trained to learn the correct embedding weights to predict the context of a given word in the corpus text, using the Skip-Gram algorithm. Finally, each word is mapped to its weights from the embedding layer of the model and the word embedding dictionary is stored in a pickle object (i.e a python-specific object) to be accessed by the other components.

The **Gold Standard Vectorizer component** takes the gold standard summaries from the training dataset and produces a final summary vector. First, the gold standard summaries sentences are tokenized and stored as a list for each summary. Sentences are then classified into narrative and non-narrative and only narrative information is processed (punctuation removal, lowercase normalization and stop words cleaning). After cleaning the text, the system transforms the summaries corpus into a vector using the word embeddings generated by the Fin2Vec component. To perform this vectorization of the gold standard summaries the first step is to transform each word into an embedding vector of 300 dimensions and add them up for each sentence to compute the vectors of the sentences. Subsequently, all sentence vectors of each summary are added together to obtain the summary vectors and the same operation is applied again to the latter to compute the final vector, which captures the relevant narrative information from the gold standard summaries. To finish, the final summary vector is stored for its further use by the Summarizer.

The **Summarizer** is the main component of the system, as its objective is to summarize a single report at a time. The Summarizer pipeline is shown in Figure 1. As the system is an extractive summarizer, the initial step is to tokenize the sentences from the original report, since these will be extracted to conform the output summary.

Given the nature of the reports, that is, a combination of numerical and narrative information, in addition to their extended length and the diversity of the information structure, it is paramount to detect narrative sections and discard those that are not, in order to get an insightful representation of the text and produce high quality summaries. The Summarizer applies narrative screening before further processing the text to remove unnecessary sentences and increase efficiency. Therefore, Part-Of-Speech (POS) tagging is applied to each word and a narrative detection analysis is carried out by identifying verb tenses within the sentence. If the sentence is non-narrative it is deleted, otherwise, it is processed for further analyses.

Once the narrative screening has been carried out, some text processing techniques are performed on the sentences: punctuation removal, lowercase normalization and stop words cleaning. After that, the Summarizer loads the word embeddings object generated by the Fin2Vec component and transforms the sentences with it, thus obtaining a word embedding vector for each sentence.

As regards feature extraction, the final system has taken into consideration three different features: (a) keyword score, (b) sentence position and (c) gold standard similarity score.

(a) To compute the keyword score for each sentence, firstly a keyword list is defined, based on previous analyses conducted manually with the purpose of identifying relevant commonly used words in the gold standard summaries (these analyses were carried out with the use of TfidfVectorizer, using different n-grams range). Then, the final score is calculated by dividing the keyword occurrences in the sentence by its total number of words.

(b) As for the position score, it positively rewards sentences appearing at the beginning of the report, as the correlation between the gold standard summaries and the first sentences from the report is extremely strong.
(c) To compute the gold standard similarity score the system utilizes the final summary vector produced by the Gold Standard Vectorizer. The cosine similarity between each sentence vector in the original report and the summary vector (that contains salient gold standard information) is measured to get the relevance of the sentence. Hence, a high gold standard similarity score demonstrates the sentence contains information relevant to the final summary.

At this point, the feature scores obtained for each sentence in the previous step are transformed into a feature matrix and normalization is applied, thus normalizing the scores between zero and one. To compute the final relevance score of the sentences the values of their three feature scores are added up. Sentences are then ranked in descending order and extracted from the original sentence list prepared at the beginning, until the length of the summary is between 900 and 1000 words.

5.1. HULAT-2

HULAT-2 system approach is identical to HULAT-1, however, it employs distinct features to extract the relevance from the sentences. Therefore, the only different component between this system and the previous one is the Summarizer.

Regarding HULAT-2 Summarizer, as shown in the Figure 1, in the feature extraction step the features used to infer the relevance of the sentences have been: (a) sentence position, (b) named entities occurrences and (c) gold standard similarity score. This approach takes advantage of the Topic Extraction Application Programming Interface (API) provided by MeaningCloud[1], which detects and extracts relevant entities from the text using complex NLP techniques. The Summarizer requests MeaningCloud API to extract the entities given the report text as input, which generates a list of entities ordered by priority.

In order to capture the most relevant information from the report and increase the efficiency of the system, only the first fifty sentences are evaluated by the Topic Extraction API. This decision has been taken after performing multiple analyses on the reports by extracting entities for different ranges of sentences. The results showed that the ten highest priority entities extracted were virtually identical for a sentence range greater than the first fifty sentences, which meant that the first lines already contained the relevant entities, dates and concepts of the entire report.

The advantage of integrating the Topic Extraction API is that it extracts different entities relevant to each report, such as company name, dates, executive names, etc., providing more accurate and important information relative to the report being analyzed, as opposed to the fixed keywords used in HULAT-1.

6. Integrated resources

To create the systems the employed resources have been: NLTK library (Bird et al., 2009) for POS tagging, lemmatization and tokenization of words and sentences (using Wordnet, perceptron and punkt module); spaCy library for the stop words list as it is more complete than other libraries (Honnibal and Montani, 2017); Sklearn library for text vectorization with the TfidfVectorizer module and cosine similarity metrics (Pedregosa et al., 2011; Buitinck et al., 2013); and PyTorch to design and train the model for the word embeddings (Paszke et al., 2019). MeaningCloud Topic Extraction tool has been used to detect entities as features of documents. Google Collab and a GPU Nvidia GeForce RTX 2080, driver version 430.5, CUDA 10.1 in Ubuntu 18.04 provided by the UC3M were used.

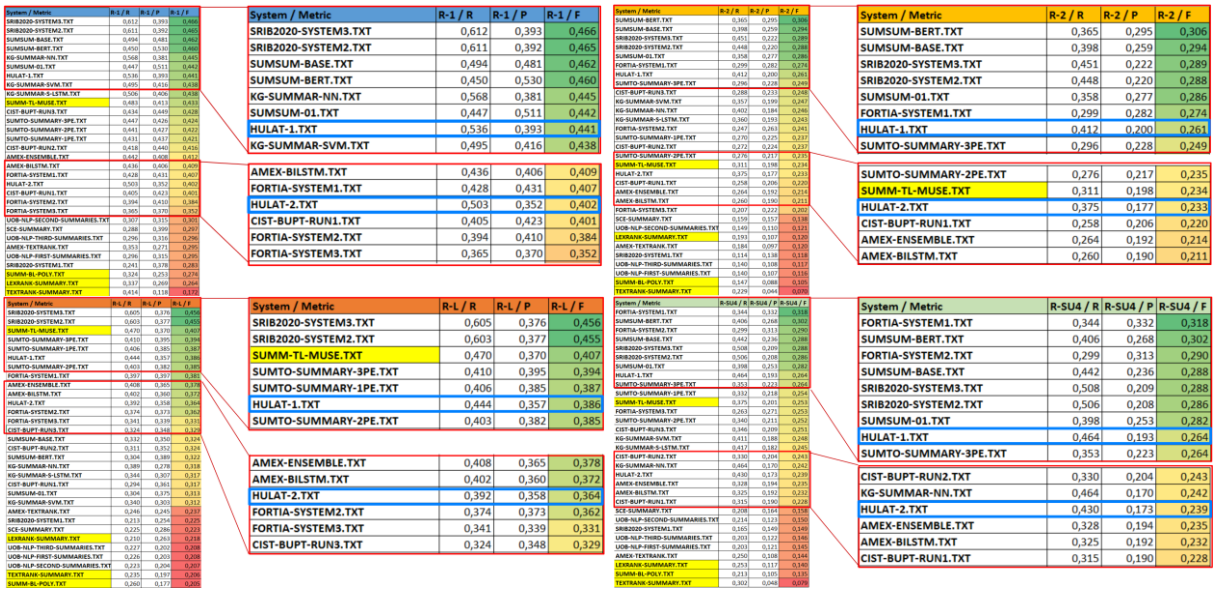
7. Results

The FNS results are shown in Figure 2, where the systems highlighted in blue are the HULAT summarizers and the yellow ones are those proposed by the task organizers. As can be appreciated, HULAT-1 is among the eight best systems submitted for the task (from 31 runs), which outperforms the

[1] www.meaningcloud.com

topline system proposed by the FNS (SUMM-TL-MUSE) except for the ROUGE-L metric. On the other hand, although HULAT-2 has achieved lower results, it still performs above almost half of all the other systems, what demonstrates its outstanding ability to identify relevant entities from each financial report, thus making it an interesting approach.

The difference in performance between the two systems is due to the list of keywords extracted from the gold standard summaries (using the TfidfVectorizer for different n-grams range), which captures more salience information than the named entities provided by the Topic Extraction API, probably because the dictionaries behind the tool are general purpose resources not adapted to the financial domain.



System / Metric	R-1 / R	R-1 / P	R-1 / F
SRIB2020-SYSTEM3.TXT	0,612	0,393	0,466
SRIB2020-SYSTEM2.TXT	0,611	0,392	0,465
SUMSUM-BASE.TXT	0,494	0,481	0,462
SUMSUM-BERT.TXT	0,450	0,530	0,460
KG-SUMMAR-NN.TXT	0,568	0,381	0,445
SUMSUM-01.TXT	0,447	0,511	0,442
HULAT-1.TXT	0,536	0,393	0,441
KG-SUMMAR-SVM.TXT	0,495	0,416	0,438
AMEX-BILSTM.TXT	0,436	0,406	0,409
FORTIA-SYSTEM1.TXT	0,428	0,431	0,407
HULAT-2.TXT	0,503	0,352	0,402
CIST-BUPT-RUN1.TXT	0,405	0,423	0,401
FORTIA-SYSTEM2.TXT	0,394	0,410	0,384
FORTIA-SYSTEM3.TXT	0,365	0,370	0,352

System / Metric	R-2 / R	R-2 / P	R-2 / F
SUMSUM-BERT.TXT	0,365	0,295	0,306
SUMSUM-BASE.TXT	0,398	0,259	0,294
SRIB2020-SYSTEM3.TXT	0,451	0,222	0,289
SRIB2020-SYSTEM2.TXT	0,448	0,220	0,288
SUMSUM-01.TXT	0,358	0,277	0,286
FORTIA-SYSTEM1.TXT	0,299	0,282	0,274
HULAT-1.TXT	0,412	0,200	0,261
SUMTO-SUMMARY-3PE.TXT	0,296	0,228	0,249
SUMTO-SUMMARY-2PE.TXT	0,276	0,217	0,235
SUMM-TL-MUSE.TXT	0,311	0,198	0,234
HULAT-2.TXT	0,375	0,177	0,233
CIST-BUPT-RUN1.TXT	0,258	0,206	0,220
AMEX-ENSEMBLE.TXT	0,264	0,192	0,214
AMEX-BILSTM.TXT	0,260	0,190	0,211

System / Metric	R-L / R	R-L / P	R-L / F
SRIB2020-SYSTEM3.TXT	0,605	0,376	0,456
SRIB2020-SYSTEM2.TXT	0,603	0,377	0,455
SUMM-TL-MUSE.TXT	0,470	0,370	0,407
SUMTO-SUMMARY-3PE.TXT	0,410	0,395	0,394
SUMTO-SUMMARY-1PE.TXT	0,406	0,385	0,387
HULAT-1.TXT	0,444	0,357	0,386
SUMTO-SUMMARY-2PE.TXT	0,403	0,382	0,385
AMEX-ENSEMBLE.TXT	0,408	0,365	0,378
AMEX-BILSTM.TXT	0,402	0,360	0,372
HULAT-2.TXT	0,392	0,358	0,364
FORTIA-SYSTEM2.TXT	0,374	0,373	0,362
FORTIA-SYSTEM3.TXT	0,341	0,339	0,331
CIST-BUPT-RUN3.TXT	0,324	0,348	0,329

System / Metric	R-SU4 / R	R-SU4 / P	R-SU4 / F
FORTIA-SYSTEM1.TXT	0,344	0,332	0,318
SUMSUM-BERT.TXT	0,406	0,268	0,302
FORTIA-SYSTEM2.TXT	0,299	0,313	0,290
SUMSUM-BASE.TXT	0,442	0,236	0,288
SRIB2020-SYSTEM3.TXT	0,508	0,209	0,288
SRIB2020-SYSTEM2.TXT	0,506	0,208	0,286
SUMSUM-01.TXT	0,398	0,253	0,282
HULAT-1.TXT	0,464	0,193	0,264
SUMTO-SUMMARY-3PE.TXT	0,353	0,223	0,264
CIST-BUPT-RUN2.TXT	0,330	0,204	0,243
KG-SUMMAR-NN.TXT	0,464	0,170	0,242
HULAT-2.TXT	0,430	0,173	0,239
AMEX-ENSEMBLE.TXT	0,328	0,194	0,235
AMEX-BILSTM.TXT	0,325	0,192	0,232
CIST-BUPT-RUN1.TXT	0,315	0,190	0,228

Figure 2. FNS 2020 results evaluated with the ROUGE 2.0 package (HULAT-1 and HULAT-2 are our summarizers)

8. Conclusion

To conclude, the extractive systems proposed in this paper offer an effective method of summarizing financial reports taking into consideration a set of features that measure the relevance of the sentences based on named entities, gold standard knowledge and sentence position. As for future enhancements, topic clustering could be performed to identify different topic sections within the report and extract the most relevant sentences from each subsection. Furthermore, it is worth noting that the performance of the systems has been limited by the reports' format, as paragraph tokenization has been impossible to perform since sentences were separated by new line characters.

Acknowledgements

This work has been partially supported by DeepEMR project TIN2017-87548-C2-1-R.

References

Adam Paszke et al., 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach et al., eds. *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035.

Alexander R. Fabbri, Irene Li, Tianwei She, Suyi Li and Dragomir R. Radev, 2019. Multi-news: A large-scale multi-document summarisation dataset and abstractive hierarchical model. *Computing Research Repository,* arXiv:1906.01749.

Ashish Vaswani, et al., 2017. Attention is all you need. *Computing Research Repository.* arXiv:1706.03762.

El-Haj, M., Rayson, P., Bouamor, H., Giannakopoulos, G., Litvak, M., AbuRaed, A., Mariko, D., Valsamou, D., Ferradans, S, Athanasakou, V., El Maarouf, I., Bentabet, N., Juge, R., Pittaras, N., Elhag, A., Gupta, A., Abi-Akl, H., Mazancourt, H., and Salzedo, C., 2020. Proceedings of The 1st Joint Workshop on Financial Narrative Processing and MultiLing Financial Summarisation (FNP-FNS 2020). In *Proceedings of the 28th International Conference on Computational Linguistics (COLING'2020)*, Barcelona, Spain.

Fabian Pedregosa et al., 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825−2830.

Karen Spärck Jones, 2007. Automatic summarising: The state of the art. *Information Processing & Management*, 43(6):1449-1481.

Lars Buitinck et al., 2013. API design for machine learning software: experiences from the scikit-learn project. *Workshop: Languages for Data Mining and Machine Learning*, 108–122.

Luca Cagliero and Moreno La Quatra, 2020. Extracting highlights of scientific articles: A supervised summarisation approach. *Expert Systems with Applications*, 113659.

Matthew Honnibal and Ines Montani, 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

Quanzhi Li and Qiong Zhang, 2020. Abstractive Event Summarisation on Twitter. In *Companion Proceedings of the Web Conference 2020*, 22-23.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre and Bing Xiang, 2016. Abstractive text summarisation using sequence-to-sequence RNNs and beyond. *Computing Research Repository.* arXiv:1602.06023.

Remi Juge, Imane Bentabet and Sira Ferradans, 2019. The fintoc-2019 shared task: Financial document structure extraction. In *Proceedings of the Second Financial Narrative Processing Workshop (FNP 2019)*, 51-57.

Sepp Hochreiter and Jürgen Schmidhuber, 1997. Long Short-Term Memory. *Neural Computation*, 9:1735-1780.

Steven Bird, Ewan Klein and Edward Loper, 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. Beijing: O'Reilly Media, Inc.

Sukriti Verma and Vagisha Nidhi, 2019. Extractive Summarisation using Deep Learning. *Computing Research Repository,* arXiv:1708.04439. version 2.

Wei Xu, Ralph Grishman, Adam Meyers and Alan Ritter, 2013. A preliminary study of tweet summarisation using information extraction. In *Proceedings of the Workshop on Language Analysis in Social Media*, 20-29.