Expert Systems With Applications 129 (2019) 200–215

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

SummCoder: An unsupervised framework for extractive text summarization based on deep auto-encoders

Akanksha Joshi[a,b,c,*], E. Fidalgo[a,b], E. Alegre[a,b], Laura Fernández-Robles[a,b]

[a] Universidad de León, Spain
[b] INCIBE (Spanish National Institute of Cybersecurity), León, Spain
[c] Centre for Development of Advanced Computing (CDAC), India

ARTICLE INFO

Article history:
Received 22 August 2018
Revised 26 March 2019
Accepted 27 March 2019
Available online 30 March 2019

Keywords:
Extractive text summarization
Auto-encoder
Deep learning
Sentence embedding
TOR darknet
Extractive summarization

ABSTRACT

In this paper, we propose SummCoder, a novel methodology for generic extractive text summarization of single documents. The approach generates a summary according to three sentence selection metrics formulated by us: sentence content relevance, sentence novelty, and sentence position relevance. The sentence content relevance is measured using a deep auto-encoder network, and the novelty metric is derived by exploiting the similarity among sentences represented as embeddings in a distributed semantic space. The sentence position relevance metric is a hand-designed feature, which assigns more weight to the first few sentences through a dynamic weight calculation function regulated by the document length. Furthermore, a sentence ranking and a selection technique are developed to generate the document summary by ranking the sentences according to the final score obtained through the fusion of the three sentences selection metrics. We also introduce a new summarization benchmark, Tor Illegal Documents Summarization (TIDSumm) dataset, especially to assist Law Enforcement Agencies (LEAs), that contains two sets of ground truth summaries, manually created, for 100 web documents extracted from onion websites in Tor (The Onion Router) network. Empirical results show that, on DUC 2002, on Blog Summarization, and on TIDSumm datasets, our text summarization approach obtains comparable or better performance than the state-of-the-art methods for different ROUGE metrics.

1. Introduction

With the advent of the Internet and a massive amount of textual data available, the immediate challenge is to develop new tools to represent the content in a concise form called summary. Automatic text summarization is an important branch of natural language processing that aims to represent long text documents in a compressed way so that the information can be quickly understood and readable for end users. We can broadly classify text summarization techniques into two categories; extractive and abstractive text summarization (Gambhir & Gupta, 2017). Extractive text summarization concatenates the most relevant sentences from the document to produce the summary. It usually comprises of three major steps: intermediate representation of input text, sentence scoring, and sentence selection. Conversely, abstractive text summarization generates the summary by paraphrasing the main contents of the document using natural language generation techniques. The summarization can also be categorized as single-document and multi-document depending on the number of input documents given (Zajic, Dorr, & Lin, 2008; Fattah & Ren, 2009). Similarly, the summarization can be either generic or query-based (Gong & Liu, 2001; Dunlavy, O'Leary, Conroy, & Schlesinger, 2007; Wan, 2008; Ouyang, Li, Li, & Lu, 2011). The generic summarization provides an overall idea of the document content whereas, query-focused summarization presents the relevant content of the document according to the user given queries. In this paper, our focus is on the task of generic extractive text summarization on single documents.

Various traditional methods for extractive text summarization proposed in the literature are based mainly on human-engineered features, as for example, combination of statistical and linguistic features such as term frequency (Luhn, 1958; Nenkova & Vanderwende, 2005), sentence length and position (Erkan & Radev, 2004) or cue and stigma words (Edmundson, 1969). In these methods, a score is assigned to each sentence based on its features. To select sentences to generate a summary, various techniques have been proposed, including greedy approaches (Carbonell & Goldstein 1998), graph-based approaches (Erkan & Radev, 2004; Wan,

* Corresponding author at: Universidad de León, Spain.
E-mail addresses: ajos@unileon.es (A. Joshi), eduardo.fidalgo@unileon.es (E. Fidalgo), enrique.alegre@unileon.es (E. Alegre), l.fernandez@unileon.es (L. Fernández-Robles).

https://doi.org/10.1016/j.eswa.2019.03.045
0957-4174/

2010; Parveen, Ramsl, & Strube, 2015), and optimization based approaches (Mcdonald, 2007).

In more recent works, deep learning based methods have attained impressive accuracies in many Natural Language Processing (NLP) tasks, such as question-answering (Bordes, Chopra, & Weston, 2014), natural language understanding (Collobert et al., 2011), sentiment analysis (Santos & Gatti, 2014), text classification (Zhang, Zhao, & LeCun, 2015) and language translation (Jean, Cho, Memisevic, & Bengio, 2015). Following the success of deep learning in various NLP applications, many on-going research works are also aiming to improve the text summarization task by exploiting the capabilities of deep neural networks (Nallapati, Zhai, & Zhou, 2017; Nallapati, Zhou, & Ma, 2017; Nallapati, Zhou, Santos, Gulcehre, & Xiang, 2016; Rush, Chopra, & Weston, 2015). Deep neural networks represent the data with multiple levels of abstraction after processing over several non-linear computation-intensive layers. To learn a good and semantically meaningful representation of input data, deep learning networks have to be supplied with a huge amount of training data. Most of the deep learning based approaches, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNN) require labeled data for training the deep network architectures with millions of learnable parameters.

At present, the biggest challenge in applying supervised deep learning approaches for extractive text summarization is the unavailability of large-scale extractive summaries created manually that are needed as ground truth for training the networks. This paper addresses this shortcoming by exploiting techniques, which do not require labeled data for training, specifically an unsupervised deep learning approach based on auto-encoders and sentence embeddings. Deep auto-encoders and variational auto-encoders have been previously applied for query-based single-document text summarization (Yousefi-Azar & Hamey, 2017) but, to the best of our knowledge, this is the first time that auto-encoders are used for generic text summarization in this way. Yousefi-Azar and Hamey (2017) applied auto-encoders for query-based single-document text summarization (2017), and Alami, Ennahnahi, Ouatik, and Meknassi (2018) used variational auto-encoders for Arabic text summarization. However, these approaches are different from our approach because Yousefi-Azar and Hamey (2017) and Alami et al. (2018) trained the auto-encoders by representing the text document using term frequency-inverse document frequency (TF-IDF) vectors, which completely ignore the word order. Li, Wang, Lam, Zhaochun Ren, and Bing (2017)) proposed variational auto-encoder for latent semantic modeling of sentences to estimate their salience for multi-document summarization. In contrast, we trained the auto-encoder using sentence embeddings, which takes into consideration, the order of words and thus, the meaning of the sentence is better preserved. Besides that, we represent the documents in a high-level concept space through an auto-encoder, which efficiently captures the semantics of the document and explores the inter-relationship among the sentences.

The key contributions of the paper can be summarized as follows:

- We introduce SummCoder, a novel and unsupervised approach for single-document extractive text summarization. We propose three sentence quality metrics: sentence content relevance, sentence novelty and sentence position in the document. The computation of the content relevance parameter is based on an auto-encoder network trained by us and the sentence novelty is determined using sentence embeddings generated by using skip-thoughts model (Kiros et al., 2015), fine-tuned on our data. The sentence position parameter is a hand-crafted feature, which dynamically assigns weight to each sentence taking into consideration the number of total sentences in the document.
- A sentence ranking and selection strategy that is derived based on the fusion of scores obtained from the three proposed sentence features. The output summary is produced by selecting the top-ranked sentences constrained by the pre-defined length of the summary.
- To evaluate our approach more comprehensively and to verify the summarization results in a new domain, we introduce a new text summarization benchmark dataset, named Tor Illegal documents Summarization (TIDSumm). The dataset consists of web documents related to illegal hidden services from the Tor[1] (The Onion Router) Dark Net (Wood, 2010). The dataset contains documents along with two sets of human-generated ground truth summaries for 100 onion web documents.

We empirically evaluated our models on this newly created dataset, TIDSumm, together with two publicly well-known benchmark datasets, DUC 2002[2] and Blog Summarization[3] corpus. The experimental results suggest that the proposed methodology significantly outperforms traditional approaches and it is better or comparable to the state-of-the-art methods for extractive text summarization.

Through the results on test datasets from different domains (i.e. news, illegal web documents and blogs), we validate the generalization capabilities of our proposed approach. The primary objective for creating TIDSumm dataset was to investigate summarization of illegal content seized by Law Enforcement Agencies (LEAs) quite frequently. The discussions with Spanish law enforcement officials revealed that LEAs investigate a vast amount of text documentation that must be individually reviewed, processed, categorized and sometimes assigned to the corresponding department, e.g. counterfeit or terrorism. Most of the time, the datasets of documents to be scanned are large, and the percentage of relevant documents is quite low in it. As time is a highly limited factor of a human investigator, through automatic summarization of documents on TIDSumm dataset, we demonstrate how text document summarization could support cyber forensic investigators to find significant information faster. In some scenarios, even if legal constraint requires reading the complete document, the proposed approach is still advantageous as it can be utilized for first level screening of documents through their generated summaries, followed by a complete revision of the relevant documents. TIDSumm was created in a similar way to DUC 2002 standard, but it contains documents from a different domain. We also made TIDSumm publicly available on our web page[4] to provide other researchers with the opportunity of evaluating their extractive summarization approaches on another dataset apart from DUC 2002.

The rest of the paper is organized as follows. Section 2 reviews the related work. The description of the proposed approach is given in Section 3. Section 4 discusses the datasets used, the experimentation carried out and the obtained results. In Section 5, we summarize our work and the main contributions, proposing some ideas to extend this work in the future.

2. Related work

Our work focuses on proposing a new approach for unsupervised text summarization based on deep learning, which utilizes auto-encoders and sentence embeddings, apart from sentence novelty and a hand-crafted feature, sentence position. For this reason, we will briefly review some learning based summarization approaches, some others related to sentence representation,

[1] https://www.torproject.org/about/overview.html.en.
[2] http://www-nlpir.nist.gov/projects/duc/guidelines/2002.html.
[3] http://www.ntu.edu.sg/home/axsun/datasets.html.
[4] http://gvis.unileon.es/dataset/tidsumm/.

especially the ones used to generate sentence embeddings and some related to hand-crafted features. Methods to obtain generic extractive summaries usually rank the sentences of a document and select a subset of them to produce a concise and fluent summary.

In the literature, a vast corpus of extractive summarization related articles is available involving both supervised and unsupervised methods. The supervised learning based approaches (Cheng & Lapata, 2016; Fattah & Ren, 2009; Kaikhah, 2004; Li, Zhou, Xue, Zha, & Yu, 2009; Ouyang et al., 2011; Svore, Way, Vanderwende, & Burges, 2007; Nallapati, Zhai, & Zhou, 2017) require gold summaries at the training phase. In contrast, unsupervised systems (Alguliev & Aliguliyev, 2008; Dunlavy et al., 2007; Fattah & Ren, 2009; Wang, Li, Zhu, & Ding, 2008; Wan, 2010; Fang, Mu, Deng, & Wu, 2017; Parveen et al., 2015) mostly rank sentences using some heuristics and they do not require manually created gold summaries for training (Yousefi-Azar & Hamey, 2017). Due to this, unsupervised methods can be easily adapted to new domains without much alteration. The earlier methods of unsupervised text summarization mainly rely on hand-crafted features such as term frequency as Luhn (Luhn, 1958); SumBasic (Nenkova & Vanderwende, 2005), sentence position and length as LexRank (Erkan & Radev, 2004) or cue and stigma words (Edmundson, 1969). The selected features in these approaches are used to obtain a ranking score for each sentence. Some approaches used graph-based ranking such as TextRank (Mihalcea & Tarau, 2004) and LexRank (Erkan & Radev, 2004). In another well-known classical approach, Latent Semantic Analysis (Steinberger & Jezek, 2004) finds salient sentences in the document by applying Singular Value Decomposition (SVD). In KLSum (Haghighi & Vanderwende, 2009), the authors proposed to generate a summary based on the criteria of Kullback-Lieber (KL) divergence.

In more recent works, some methods based on supervised and unsupervised deep learning techniques have been proposed for text summarization. Zhong, Liu, Li, and Long (2015) proposed an unsupervised deep architecture based on Restricted Boltzmann Machines to perform query-oriented multi-document summarization. Yousefi-Azar and Hamey (2017), proposed an unsupervised query-oriented extractive text summarization approach based on deep auto-encoders using term-frequency (TF) features. They also introduced an ensemble of noisy auto-encoders that adds noise to the input vectors and selects the top-ranked sentences from an ensemble of noisy runs. Among the supervised deep learning methods, Cao, Wei, Dong, Li, and Zhou (2015) proposed a recursive neural network for salient sentence ranking in multi-document summarization. Denil, Demiraj, and de Freitas (2015) introduced a hierarchical CNN architecture to support introspection of the document structure. Cao et al., (2015) derived summary prior features using enhanced CNNs for the task of extractive text summarization. Cheng and Lapata (2016), applied a supervised neural attention model to select the significant sentences and words in the document to generate an extractive summary. Cao, Li, Li, Wei, and Li (2016) proposed a query-focused summarization system termed as “Attsum” based on CNNs which jointly learns sentence saliency and sentence relevance ranking. Their approach automatically learns the distributed representations of sentences and documents and applies the attention mechanism to simulate human behavior when a query is given. Nallapati, Zhou, and Ma (2017) proposed an RNN approach based on sequence model and provided a methodology to achieve extractive text summarization. Wu and Hu (2018) designed Reinforce Neural Extractive summarizer (RNES) model to generate a coherent and informative summary simultaneously. (Narayan, Cohen, & Lapata, 2018) conceptualize extractive summarization by optimizing the ROUGE evaluation metric using reinforcement learning. Liu, Saleh, Pot, Goodrich, Sepassi, Kaiser, and Shazeer (2018) generated Wikipedia articles by projecting the problem as a multi-document summarization task and utilized decoder only networks for achieving summarization. Mehta, Arora, and Majumder (2018), have proposed a new summarization technique based on context embedding for determining the focus of the article and jointly used attention based LSTM networks to select important sentences from scientific articles for summarization.

In this work, we have utilized unsupervised deep auto-encoders, which have already been successfully applied for various image processing tasks. However, their capabilities for NLP related tasks have not been explored much. Genest, Gotti, and Bengio (2011) developed an auto-encoder based supervised and unsupervised approach to evaluate human generated automatically and system generated summaries. Kumar and D'Haro (2015) developed Deep Auto-encoder Topic Model (DATM) approach for the identification of topics from short texts. Recently, various variants of auto-encoders have also been proposed such as denoising auto-encoder (Vincent, Larochelle, Lajoie, Bengio, & Antoine Manzagol, 2010), contractive auto-encoder (Rifai, Vincent, Muller, Glorot, & Bengio, 2011), variational auto-encoder (Kingma & Welling, 2013), and k-sparse auto-encoder (Makhzani & Frey, 2014) for various text-related tasks.

Now, we briefly discuss some approaches for the sentence or document representation in a machine-understandable format, which is one of the critical aspects in the learning mechanism for automatic document summarization. Sentences usually are taken as the basic unit to be selected for a summary generation because they are considered as the smallest grammatical unit which can represent a complete statement (Ceylan, Mihalcea, Öyertem, Lloret, & Palomar, 2010). During the summarization phase, the sentences are represented as vectors to be used as an input to the algorithm. In most of the traditional text summarization approaches (Erkan & Radev, 2004; Mihalcea & Tarau, 2004) and even in some recent ones (Wan, 2010; Fang et al., 2017; Parveen et al., 2015), researchers have used hand-crafted features to represent the sentences in the document. Recently, authors have presented various methods to learn embedding vectors for sentences, which involves training a model that can automatically map a sentence to a vector that encodes the semantic meaning of the sentence (Socher, Perelygin, & Wu, 2013; Kim, 2014; Pagliardini, Gupta, & Jaggi, 2017; Palangi et al., 2016). Sentence embeddings are an effective way of representing the input document as compared to the traditional text representation methods such as TF-IDF because they take into account the order of words and surrounding context to capture the semantic similarities between text phrases correctly. Researchers have investigated various approaches to generate sentence embeddings including supervised methods, which learn representation for some specific task such as sentiment analysis and sentence classification. Some popular approaches in this category include recursive networks (Socher et al., 2013), RNNs (Hochreiter & Schmidhuber, 1997), and CNNs (Kim, 2014; Kalchbrenner, Grefenstette, & Blunsom, 2014). Moreover, there are some unsupervised methods proposed for producing sentence representations (Pagliardini, Gupta, & Jaggi, 2017; Palangi et al., 2016) such as skip-thought vectors (Kiros et al., 2015) and paragraph vector (Le & Mikolov, 2014), which can be used for various purposes like image-sentence ranking, paraphrase detection, and semantic relatedness. For our work, we employed skip-thought vectors (Kiros et al., 2015) to generate sentence embeddings as skip-thought vectors are generated using an unsupervised approach and can be fine-tuned for news dataset without any supervision.

Some text summarization approaches have been proposed in the literature, which utilizes sentence embeddings. Kageback, Mogren, Tahmasebi, and Dubhashi (2014) used continuous vector space models to achieve extractive text summarization based on submodular optimization (Lin & Bilmes, 2011). Kobayashi, Yatsuka,

and Taichi (2015) and Yogatama and Smith (2015) employed pre-trained sentence embeddings to generate summaries using unsupervised optimization techniques. Yin and Pei (2015) used CNNs to project sentences into densely distributed representations and then applied a diversified selection technique to select salient sentences of the document. Ma, Deng, and Yang (2016) proposed paragraph vectors to reconstruct the document from the summary through a neural document model for multi-document summarization. Yasunaga et al. (2017) generated multi-document summaries by applying graph-based CNNs which take sentence embeddings obtained using RNNs as input node features. Zhang et al. (2017) introduced a sentence vector encoding framework to encode sentences based on encoder-decoder networks and used it for single line text summarization. Sinha, Yadav, and Gahlot (2018) presented a data-driven approach towards extractive text summarization which uses sentence embeddings and chooses sentences according to the trained neural networks with cross-entropy loss.

Earlier, research on text summarization used statistical and linguistic features (Fattah & Ren, 2009; Ko & Seo, 2008) for representing text documents such as position of sentences, positive and negative keywords, sentence centrality, resemblance of sentences to the title, relative length of sentences or presence of numerical data in a sentences. Among them, sentence position is deemed as one of the essential features that contribute towards the accuracy of text summarization systems (Yeh, Ke, Yang, & Meng, 2005). According to Gupta, Pendluri, and Vats (2011) and Abuobieda, Salim, Albaham, Osman, and Kumar (2012), the first sentence of a paragraph is important and a strong candidate for summary generation. In our approach, we also take the position of the sentence into account so that the sentences, which occur in the first few lines, get higher weights as compared to those that appear posterior in the document.

Another critical aspect of text summarization is to minimize redundancy in the output summary. Various approaches have been introduced in the literature, which handles redundant content while summary generation (Sarkar, 2010; Carbonell & Goldstein, 1998; Huang, He, Wei, & Li, 2010; Lloret & Palomar, 2013). In our proposed framework, we calculate the novelty score of each sentence based on the similarity between sentence embeddings to handle the redundancy while creating document summaries.

Some methods for text summarization, published in the last years share the same objective or some ideas with the one we are proposing. Next, we present the main differences between our method and some of the closest ones.

One of the main differences is that our proposal utilizes deep auto-encoders which are trained in an unsupervised manner, instead of using gold summaries for training. In this respect, our method differs from supervised approaches, as for example Cao, Wei, Dong, et al. (2015), Denil et al. (2015), Cao et al. (2015), Cao, Li, et al. (2016), Cheng and Lapata (2016), Nallapati, Zhou, and Ma (2017), Nallapati, Zhai, and Zhou (2017), Narayan et al. (2018), Wu and Hu (2018), Zhang et al. (2017), Liu et al. (2018), Mehta et al. (2018), and Sinha et al. (2018).

Our approach also differs from unsupervised systems proposed in the literature. Yousefi-Azar & Hamey, 2017 also utilized auto-encoder, but their method differs firstly in that they considered as input term-frequency (TF), after adding small random noise to local TF and they termed their auto-encoder networks Ensemble Noisy Auto-Encoder (ENAE). In our case, we use a set of sentence embeddings which are better representations than TF features (Le & Mikolov, 2014) as input to our auto-encoder model. Second, we also considered sentence position and sentence novelty measure, and lastly, they proposed their approach for query oriented single-document text summarization rather than generic extractive summarization, as our method does. Other related work is Alami et al. (2018) who developed a summarization method that provides a latent sentence representation using variational auto-encoders but, opposed to our approach, they did not explore any relationship between the sentences of a document.

Moreover, they specifically applied it for Arabic text summarization rather than evaluating it on a standard summarization dataset. Yin and Pei (2015) used CNN to generate sentence representation/embeddings and then they applied a diversified selection process (DivSelect) to select salient sentences of the document. In our work sentence representation is obtained through recurrent neural networks (i.e. skip-thoughts) which are trained on a large dataset, Book Corpus (Zhu et al., 2015) what gives a better sentence representation. On top of that, we applied auto-encoders to explore the inter-relationship among document sentences, but Yin and Pei's (2015) work fails to explore any such relation between sentences in the document and does not incorporate sentence position in their approach.

It should also be noted that our utilization of sentence position is different from previous works (Edmundson, 1969; Erkan & Radev, 2004; Yeh et al., 2005) as we have developed a new algorithm which assigns position score dynamically and provides higher weights automatically only to few starting sentences based on relative sentence position and document length. However, in most of the previous methods relative position of a sentence in the document is generally assigned as a weight to that sentence in a fixed manner or through some function (e.g. linearly decreasing function). Due to this, the position scores of sentences appearing later in the document is always lesser as compared to former ones, which makes them less significant. On the other hand, our approach, after a few starting sentences, assigns the constant weights for other sentences in the document which makes them equally important.

3. Proposed SummCoder framework

The overall pipeline of the proposed SummCoder framework is illustrated in Fig. 1. In the following section, we describe the key components of the proposed deep learning framework for extractive summary generation.

3.1. Preprocessing

Firstly, we pre-processed each document before passing it to the next stage of the sentence encoder. We extracted text from the documents by removing any XML/HTML tags and metadata and then tokenized the text into sentences. We cleaned the document by removing special characters, redundant whitespaces, numbers, URLs and email addresses. All the words were converted to lower case and sentences containing less than five words were omitted while training the network. For pre-processing, we used Natural Language ToolKit[5] (NLTK) library and regular expressions.

3.2. Sentence encoder (vector representation)

After pre-processing (Section 3.1) of the input document, the next step is to map each sentence into a fixed-length vector of real numbers, and we used skip-thoughts model proposed by Kiros et al. (2015) to obtain sentence embeddings. It is an unsupervised approach (the model is termed as "skip-thoughts") to train a generic, distributed sentence encoder which models sentences into vectors called skip-thought vectors. The skip-thoughts model is inspired by word vectors learning (Mikolov, Chen, Corrado, & Dean, 2013), abstracted to the sentence level in the sense that sentence vectors are learned using surrounding sentences. It considers that

[5] http://www.nltk.org/.

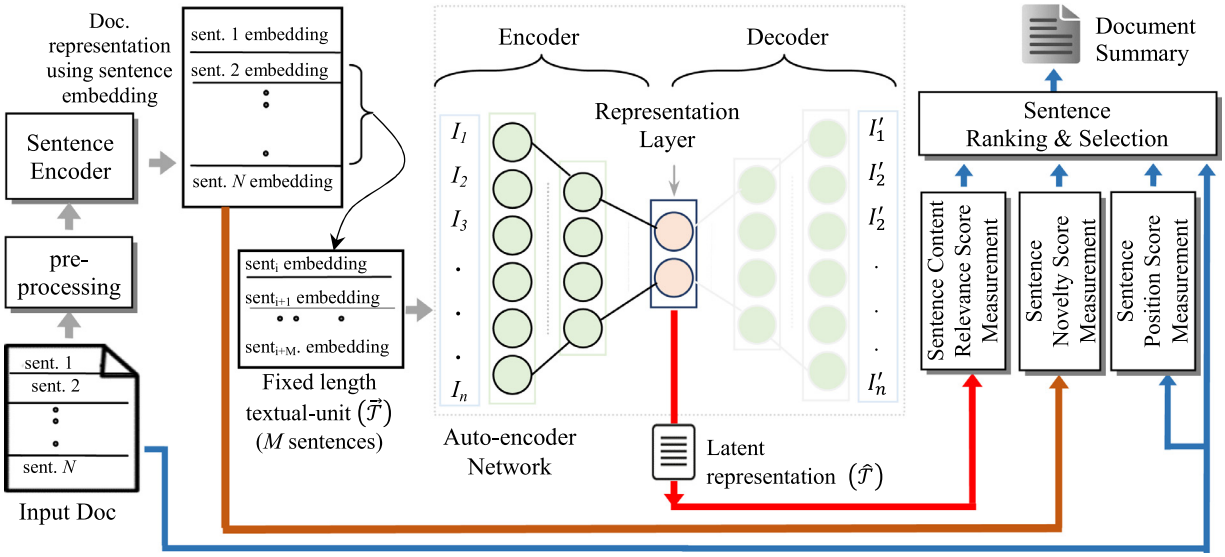

Fig. 1. Illustration of Proposed SummCoder Framework. The Sentence Encoder generates sentence embedding for each sentence to represent input document embedding. The input to auto-encoder network is fixed length textual-units created from document embedding. Sentence Content Relevance Score is computed using latent representation learnt through auto-encoder model. Sentence Novelty Score is obtained directly from sentence embeddings in document and the Sentence Position Score directly from input document. Sentence Ranking & Selection module fuses the 3 scores and computes the final ranks for each sentence to select high ranking sentences for the final document summary.

nearby sentences provide rich semantic and contextual information. The model is based on encoder-decoder networks such as an RNN encoder with Gated Recurrent Units (GRUs) (Chung, Gulcehre, Cho, & Bengio, 2014) activations and an RNN decoder with conditional GRUs. The purpose of an encoder network is to map an English sentence into a vector and then the decoder part tries to map encoded vector into surrounding sentences. The encoder-decoder architecture pushes the model to select and abstract some important features and creates the connection between a source and a target. The complete network is trained to reconstruct the surrounding sentences to map sentences, which have semantic and syntactic similarity into identical vector representations. The unidirectional skip-thoughts model has been trained on unlabeled BookCorpus dataset (Zhu et al., 2015) and achieved state-of-the-art accuracies on various tasks such as semantic relatedness, paraphrase detection, image-sentence ranking, or classification.

3.3. Summary generator

The extractive summarization problem can be defined as a sentence sequence ranking or sentence selection problem in a document. To determine which sentences should be included in the summary, we propose three criteria:

i. Sentence Content Relevance Metric ($score^{ContR}$)
ii. Sentence Novelty Metric ($score^{Nov}$)
iii. Sentence Position Relevance Metric ($score^{PosR}$)

The proposed framework ranks sentences based on these three metrics. Formally, we consider a document $\mathcal{D}$ as a bag-of-passages or a bag-of-textual-units and each textual unit as a bag-of-sentences. Let a textual unit be denoted as $\mathcal{T} \in \mathcal{D}$, and a sentence as $\mathcal{S} \in \mathcal{T}$ or $\mathcal{S} \in \mathcal{D}$. Therefore, a document can be considered as a bag-of-sentences. Let there are N sentences in the document $\mathcal{D} = (\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_N)$, and a textual unit can have M sentences, $\mathcal{T} = (\mathcal{S}_1, \mathcal{S}_2 \ldots, \mathcal{S}_M)$ $|1 \leq M \leq N$.

A sentence embedding stage precedes the computation of sentence novelty and sentence content relevance metrics. We sequentially process each sentence in the document and obtain its corresponding embedding vector using the fine-tuned skip-thoughts model as explained in Section 3.2. The sentence embedding denoted as $\vec{\mathcal{S}}$, is a real-valued vector in an n-dimensional Euclidean space $\mathbb{R}^n$. The major benefit of embedding vector representation is that, the sentences with similar meaning are mapped to similar vector representations and simultaneously, sentences of different meanings are mapped to different vector representations. In the following subsections, we explain the details for the computation of the three novel criteria and the final sentence ranking and selection criteria.

3.3.1. Sentence content relevance metric

One of the objectives in our proposal is to learn an appropriate relevance score for each sentence in the document. Our approach uses a lower dimensional latent representation of the document to compute sentence relevance scores. We assume that the sentences, which contain more significant keywords and phrases, can be considered more important as compared to other sentences in the document. So our main intent is to derive a method to discover the intrinsic representation of the document in which unimportant words/phrases from the sentences/textual-units are attenuated. Since a labeled dataset is not available to train a supervised network, we adopted an unsupervised auto-encoder network-based approach.

An auto-encoder proposed by Hinton and Salakhutdinov (2006), is a particular type of feed-forward neural network designed to reduce data dimensionality. It consists of identical input and output layers and has at least one hidden layer with lower dimensions as compared to the input data. An auto-encoder consists of two parts: encoder $\xi(\bullet)$ and decoder $\delta(\bullet)$. The encoder takes as input data, x, of dimension d ($x \in R^d$) and tries to learn a lower dimensional hidden representation $h(x) \in \mathbb{R}^{d1} | d1 < d$. The decoder part learns to reconstruct the input data in its output layer, so that, $\delta(h(x)) \approx x$.

We designed an auto-encoder network (a sample network architecture utilized in this work is given in Table 2) to obtain a lower dimensional representation of the input document. In our approach, we obtained document representation by first representing each sentence $\mathcal{S}$ in the document $\mathcal{D}$ as a 2400 dimensional embedding vector, $\vec{\mathcal{S}} \in \mathbb{R}^{2400}$ and finally, combined them to generate document embeddings. Since the number of sentences in input documents may vary, we train the auto-encoder network using fixed length textual-unit embeddings as input to the auto-encoder. The textual-unit embeddings, $\vec{\mathcal{T}}$, are obtained by sequentially con-

catenating M sentence embeddings, given as:

$$\vec{\mathcal{T}}_i = \begin{cases} \left(\vec{\mathcal{S}}_{1+(i-1)M}, \vec{\mathcal{S}}_{2+(i-1)M}, .., \vec{\mathcal{S}}_{i*M}\right) \mid 1 \le i \le \lfloor N/M \rfloor \\ \left(\vec{\mathcal{S}}_{N-M+1}, \ \vec{\mathcal{S}}_{N-M+2}, .., \vec{\mathcal{S}}_N\right) \mid N \ mod \ M \rangle 0, \ i = \lfloor N/M \rfloor \end{cases} \quad (1)$$

After training the auto-encoder network, we discarded the decoder part and used only the encoder part to generate a lower dimensional representation of each textual-unit embedding. We term the lower-dimension output of the encoder as "Latent Representation". In our case, since the input to the auto-encoder are textual-unit embeddings, we name the encoder outputs as latent textual-unit representations, denoted as $\hat{\mathcal{T}}$. Furthermore, by concatenating the latent textual-unit representations of the document sequentially, we obtain the latent document representation, $\hat{\mathcal{D}}$, given as:

$$\hat{\mathcal{D}} = \left(\hat{\mathcal{T}}_1, \hat{\mathcal{T}}_2, \ldots, \hat{\mathcal{T}}_k\right) | k = \lfloor N/M \rfloor \quad (2)$$

To measure the relevance of a sentence in the document based on its content, we first compute a latent document representation i.e. obtained in (2), which we refer here as original latent representation. Next, we remove the current sentence from the input document and generate another latent representation with remaining sentences in the document, which we will refer to as modified latent representation. More precisely, to generate the modified latent representation we replace the embedding vector of the considered sentence from the input textual-unit embedding with a vector of the same size containing all zeroes and fed it to auto-encoder to obtain the modified latent document representation denoted as $mod\hat{\mathcal{D}}_{\mathcal{S}_i}$ (symbolically, $mod\hat{\mathcal{D}}_{\mathcal{S}_i} = \hat{\mathcal{D}} - \hat{\mathcal{S}}_i$).

To estimate the sentence content relevance score, $\text{score}^{\text{ContR}}(\mathcal{D}, \mathcal{S}_i)$ for each sentence ($\mathcal{S}_i$), we generate the modified latent document representation $mod\hat{\mathcal{D}}_{\mathcal{S}_i}$ for all the sentences, based on the aforementioned process and then compute the cosine similarity between the original latent document representation ($\hat{\mathcal{D}}$) and each modified latent representation ($mod\hat{\mathcal{D}}_{\mathcal{S}_i}$), defined as:

$$\text{score}^{\text{ContR}}(\mathcal{D}, \ \mathcal{S}_i) = 1 - \frac{\hat{\mathcal{D}} \cdot mod\hat{\mathcal{D}}_{\mathcal{S}_i}}{\left\|\hat{\mathcal{D}}\right\| \left\|mod\hat{\mathcal{D}}_{\mathcal{S}_i}\right\|} \quad (3)$$

The value of $\text{score}^{\text{ContR}}$ is bounded in [0,1] and a higher value represents a more relevant sentence. It is reasonable to assume that, in any document, important sentences contain a higher number of important words or phrases compared to less important sentences. Thus, when an embedding corresponding to a more significant sentence is removed from the document, the similarity score between the original and the modified latent representation would be comparatively lower than when an embedding corresponding to a less important sentence is removed from the input. The removal of the most important sentence should produce the highest dissimilarity between the original and modified latent representations. The intuition behind this approach is that the auto-encoder network automatically attenuates the unnecessary words or sentences while generating latent textual-unit representations and detects the salient words or sentences in the input textual-unit.

3.3.2. Sentence novelty metric

This parameter computes the novelty metric for each sentence in a document. The aim is to assign a low score when the sentence is redundant or repetitive and high score when it is novel. We obtain the similarity between two sentences $\mathcal{S}_i$ and $\mathcal{S}_j$ as the cosine similarity of their corresponding embedding vectors, $\vec{\mathcal{S}}_i$ and $\vec{\mathcal{S}}_j$, given as:

$$\text{Sim}\left(\vec{\mathcal{S}}_i, \vec{\mathcal{S}}_j\right) = \frac{\vec{\mathcal{S}}_i \cdot \vec{\mathcal{S}}_j}{\left\|\vec{\mathcal{S}}_i\right\| \left\|\vec{\mathcal{S}}_j\right\|} \quad (4)$$

To obtain the global novelty of a sentence $\mathcal{S}_i$ in the document, we compute its similarity with remaining sentences and if the similarity is found below a pre-determined threshold, then it is considered novel. Moreover, with this strategy, when two sentences are found similar to each other beyond the pre-determined threshold, we assign a higher novelty score to the sentence, which has a higher value for content relevance measure ($\text{score}^{\text{ContR}}$). The computation of sentence novelty metric is given as:

$$\text{score}^{\text{Nov}}(\mathcal{D}, \ \mathcal{S}_i) = \begin{cases} 1, \text{if } \max\left(\left\{\text{Sim}\left(\vec{\mathcal{S}}_i, \vec{\mathcal{S}}_j\right)\right\}\right) < \tau, 1 \le j \le N, i \ne j \\ 1, \text{if} \max\left(\left\{\text{Sim}\left(\vec{\mathcal{S}}_i, \vec{\mathcal{S}}_j\right)\right\}\right) > \tau, \text{score}^{\text{ContR}}(\mathcal{D}, \mathcal{S}_i) \\ \qquad > \text{score}^{\text{ContR}}(\mathcal{D}, \ \mathcal{S}_k) \\ k = \text{argmax}\left(\text{Sim}\left(\vec{\mathcal{S}}_i, \vec{\mathcal{S}}_j\right)\right), 1 \le j \le N, i \ne j \\ 1 - \max\left(\left\{\text{Sim}\left(\vec{\mathcal{S}}_i, \vec{\mathcal{S}}_j\right)\right\}\right), \text{Otherwise}, \end{cases} \quad (5)$$

The value of $\text{Score}^{\text{Nov}}$ parameter is in the range [0, 1], as $\text{Sim}(\vec{\mathcal{S}}_i, \vec{\mathcal{S}}_j) \in [0, 1]$ and 'τ' is the threshold and for all our experiments we used a value of τ=0.85. To decide the value of τ, we used 50 sentence pairs; 25 pairs of sentences with some semantic similarity between them and the remaining 25 pairs with no similarity (i.e. different meaning). Then the cosine similarity (Eq. (4)) between the sentence embeddings of all the pairs is computed. It was found that for pairs with similar meaning the value of $\text{Sim}(\vec{\mathcal{S}}_i, \vec{\mathcal{S}}_j)$ was in the range of [0.79, 0.99] with a mean of 0.89 and for pairs with a different meaning, it was in the range of [0.2, 0.75] with a mean of 0.68. Also, for similar sentence pairs, only four pairs out of 25, obtained a cosine similarity below 0.85 (i.e. 0.79, 0.81, 0.82, 0.84), hence we considered τ=0.85 as a good choice in the computation of $\text{score}^{\text{Nov}}(\mathcal{D}, \ \mathcal{S}_i)$.

3.3.3. Sentence position relevance metric

The sentence position is an early sentence-based heuristic for text summarization (Edmundson, 1969). We devised sentence position relevance metric, $\text{score}^{\text{PosR}}(\mathcal{D}, \ \mathcal{S}_i)$, as another important parameter which contributes to the overall sentence ranking in the document and is defined for each sentence, $\mathcal{S}_i$, as:

$$\text{score}^{\text{PosR}}(\mathcal{D}, \mathcal{S}_i) = \max\left(0.5, \exp\left(\frac{-\mathcal{P}(\mathcal{S}_i)}{\sqrt[3]{N}}\right)\right) \quad (6)$$

where, max δ (•) is the maximum value selection function, exp (•) is the exponential function and N is the number of sentences in the document. The $\mathcal{P}(\mathcal{S}_i)$ function provide the relative position of the i^{th} sentence $\mathcal{S}_i$ in the document, given that $\mathcal{D} = (\mathcal{S}_1, \mathcal{S}_2, \ldots, \ \mathcal{S}_N)$ starting from $\mathcal{P}(\mathcal{S}_i) = 1$ for the first sentence and so on. The value of the proposed parameter $\text{score}^{\text{PosR}}$ is bounded in the range [0.5, 1.0]. The sentence position relevance score is higher for sentences located at the beginning of the document and it gets, stable at a value of 0.5 after a given number of sentences depending on the total number of sentences of the document N. The parameter gets stable earlier for documents with a few number of sentences. For example, the sentences in a 10 sentences document would get the following scores: 0.63, 0.5 .. 0.5, and for a document with 100 sentences, the scores would be: 0.81, 0.65, 0.52, 0.5 .. 0.5, and the scores for 1000 sentences: 0.90, 0.82, 0.74, 0.67, 0.61, 0.55, 0.5 .. 0.5.

3.2.4. Sentence ranking and selection

We define the final sentence score of a sentence $\mathcal{S}_i$ in a document $\mathcal{D}$, $\text{score}^{\text{f}}(\mathcal{D}, \ \mathcal{S}_i)$, using weighted score fusion of the three criteria, i.e. sentence content relevance score (Eq. (3)), sentence novelty score (Eq. (5)), and sentence position relevance score

(Eq. (6)) as follows:

$$\text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_i) = \alpha \cdot \text{score}^{\text{ContR}}(\mathcal{D}, \mathcal{S}_i) + \beta \cdot \text{score}^{\text{Nov}}(\mathcal{D}, \mathcal{S}_i) + \gamma \cdot \text{score}^{\text{PosR}}(\mathcal{D}, \mathcal{S}_i), \quad (7)$$

where, $\alpha,\ \beta, \gamma \in [0, 1]$ with $\alpha + \beta + \gamma = 1$, are fusion weights and assigns relative weights to different parameters while calculating the final sentence selection score.

Let, SCORE($\mathcal{D}$) represents an ordered list of the final sentence scores (7) of the sentences in a document $\mathcal{D}$, given as:

$$\text{SCORE}(\mathcal{D}) = \text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_1), \text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_2), \ldots., \text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_N) \quad (8)$$

Now, the relative rank of each sentence, Rank($\mathcal{S}_i$), in the document can be obtained by computing the ordinal rank of its final score in the SCORE($\mathcal{D}$) list, given as:

$$\text{Rank}(\mathcal{S}_i) = 1 + \sum_{e=1}^{N} \Psi(e, i), \text{ and}$$

$$\Psi(e, i) = \begin{cases} 1, & if\ \text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_i) + \varepsilon \cdot i > \text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_e) + \varepsilon \cdot e, \\ 0, & Otherwise \end{cases} \quad (9)$$

where, $\text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_i)$, $\text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_e) \in \text{SCORE}(\mathcal{D})$, ε(epsilon) is a tiny positive value ($\varepsilon \to 0+$) and the components with ε in (Eq. (9)) will assign a different rank value when $\text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_e) == \text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_i)$ while giving preference to the position of a sentence.

After getting the relative ranks of the sentences in a document, the next step is to select the sentences with higher ranks to generate the summary. Let the desired length (numbers of sentences) of a target summary be L. Then the target summary with L sentences, Summary($\mathcal{D}, L$), can be constructed in two ways:

A. Arranging qualified sentences in the summary in descending order of their relative ranks, given as:

$$\text{Summary}(\mathcal{D}, L) = \sum_{a=1}^{L} \mathcal{S}_i(\mathcal{D}), |\text{Rank}(\mathcal{S}_i) = a, 1 \le i \le N \quad (10)$$

B. Arranging qualified sentences in the summary in the order of their occurrence in the input document, given as:

$$\text{Summary}(\mathcal{D}, L) = \sum_{i=1}^{N} \mathcal{S}_i(\mathcal{D}), \text{if Rank}(\mathcal{S}_i) \le L \quad (11)$$

4. Experimental analysis and results

4.1. Datasets

Text summarization can be abstractive or extractive, which can be further classified as a single or multi-document based, query-oriented or generic. Depending upon different summarization tasks and problem domains, e.g. news, several datasets have been released. Some examples are CNN and DailyMail (Hermann et al., 2015), SUMMAC,[6] Gigaword,[7] TGSum (Cao, Chen, Li, Li, Wei, Zhou, et al., 2016), BBC News Summary,[8] Legal Case Reports Dataset (Galgani, Compton, & Hoffmann, 2012), WikiHow (Koupaee & Wang, 2018) or Blog Summarization dataset (Hu, Sun, & Lim, 2007, 2008), DUC datasets (Document Understanding Conferences), among others. In our context, we selected the datasets widely adopted in the literature for the problem of extractive text summarization of generic single-document. More specifically, we employed CNN and DailyMail datasets to fine-tune skip-thoughts model and training the auto-encoder network, together with DUC 2002 and Blog Summarization datasets for comparative analysis and the evaluation of the proposed approach.

We also introduce a new text summarization dataset, named Tor Illegal Documents Summarization (TIDSumm) that contains documents related to illegal activities extracted from Darknet. A summary of the datasets used for training and evaluation in this work is presented in Table 1.

4.1.1. CNN and dailymail dataset

These two datasets consist of news articles frequently used for question and answering research. The CNN contains 90,000 news stories whereas DailyMail contains 197,000 news stories. We combined both datasets to generate a larger one containing 287,000 documents. We created a holdout set of 14,350 documents for validation purposes and used the remaining 272,650 documents for unsupervised training of auto-encoder networks.

4.1.2. DUC 2002 dataset

DUC 2002 is part of Document understanding conferences and created by NIST (National Institute of Standards and Technology) for evaluation in the field of text summarization. Generic single-document summarization has been the fundamental task in DUC 2002 (e.g. task 2 in DUC 2002). For single-document extractive text summarization, DUC 2002 has 59 document sets comprising 567 news articles with approximately ten documents per category. Each document is associated with two different gold summaries with a length of around 100 words for each article.

4.1.3. Tor illegal documents summarization (TIDSumm) dataset

Motivated by the fact that Law and Enforcement Agencies daily come across millions of text content about illegal activities like drugs selling, black market or counterfeit currency, and that they need tools to track them, we introduce a new text summarization dataset, named as Tor Illegal Documents Summarization (TIDSumm) and we tested SummCoder on it. The TIDSumm dataset contains 100 documents and two gold summaries for each document written by two different persons. The nature of gold summaries is extractive, i.e. they are created by selecting representative sentences from the source document. Following the DUC 2002 benchmark, the length of summary for each document is kept close to 100 words. The standard guidelines to create gold summaries were to select diverse and novel sentences while keeping the final summary fluent and under-length limit.

The source documents in TIDSumm dataset are selected from Darknet Usage Text Addresses dataset (DUTA) (Al-Nabki, Fidalgo, Alegre & Paz, 2017), which is a publicly available dataset with 6831 documents of 26 different categories crawled over the onion web or Tor network. The onion web or Tor network is a darknet, which allows organizations and individuals to host content and access it anonymously without compromising their privacy. It comprises of web pages from the onion websites in Tor (The Onion Router) Darknet, whose content mostly refers to illegal activities. To create TIDSumm, we manually selected a subset of 100 documents from six categories of DUTA, which are mainly related to websites hosting illegal activities such as counterfeit credit card, counterfeit money, cryptocurrency, hacking, marketplace, and drugs over Tor network. The samples are selected randomly; however, while choosing, we gave preference to those documents, which have at least 250 words or more. In addition, we omitted the documents, which are redundant, and contains only URLs or images. The number of samples under different categories in TIDSumm range from 10 to 23.

4.1.4. Blog summarization dataset

This dataset (Hu et al. 2007, 2008) was created by collecting data from two blogs, Cosmic Variance and Internet Explorer Blog.

[6] http://www-nlpir.nist.gov/related_projects/tipster_summac/cmp_lg.html.
[7] https://catalog.ldc.upenn.edu/LDC2003T05.
[8] https://www.kaggle.com/pariza/bbc-news-summary.

Table 1
Description of datasets used for training and evaluation.

Dataset	Type	Usage	Number of documents	Number of categories	Length of summary
CNN	News	Training	85,500	–	–
		Validation	4500		
Dailymail	News	Training	187,150	–	–
		Validation	9850		
DUC 2002	News	Testing	567	59	100 words
Blog Summarization	Blog	Testing	100	–	7 sentences
TIDSumm (ours)	Illegal Activities	Testing	100	6	100 words

The authors randomly selected 100 documents, 50 from each blog, and they generated for each document, four gold summaries by four human annotators. Each gold summary consists of seven sentences ranked by the human summarizers.

4.2. Evaluation measures

We used Recall-Oriented Understudy for Gisting Evaluation (ROUGE) recall measure (Lin, 2004), specifically ROUGE-N (ROUGE-1, ROUGE-2), ROUGE-L, and ROUGE-SU4 to report the accuracy of our approach. DUC widely uses ROUGE for performance evaluation of summarization algorithms. We utilized ROUGE toolkit[9] (version 1.5.5) to compute results.

ROUGE-N is an N-gram recall measure between the system generated summary (also known as candidate summary) and the gold summary. It is the ratio of the count of N-gram phrases which occur in both the candidate and gold summary, to the count of all N-gram phrases that are present in the gold summary given as follows:

$$\text{ROUGE} - \text{N Recall} = \frac{\sum_{S\in(\text{Reference Summary})}\sum_{\text{N}-\text{gram}\in S}\text{Count}_{\text{match}}(\text{N}-\text{gram})}{\sum_{S\in(\text{Reference Summary})}\sum_{\text{N}-\text{gram}\in S}\text{Count}(\text{N}-\text{gram})}, \quad (12)$$

where, N is the length of N-gram, $\text{Count}_{\text{match}}$ (N-gram) is the maximum number of N-grams that occur in both gold summary and candidate summary. ROUGE-1 refers to unigram (each word) measure and ROUGE-2 refers to bigram measure between the candidate summary and gold summary. During our evaluation, we chose the best of ROUGE-N scores calculated over different reference summaries.

ROUGE-L assesses the fluency of the summary and measures the longest matching sequence of words using the Longest Common Subsequence (LCS). In LCS, the matches are not consecutive but in-sequence which reflects sentence level word order. The ROUGE-L for reference summary X of length m and candidate summary Y of length n is calculated as follows:

$$\text{ROUGE} - \text{L} = \frac{\text{LCS}(X,Y)}{m}, \quad (13)$$

where, LCS(X, Y) is the length of a longest common subsequence of X and Y.

ROUGE-SU4 counts skip-bigrams along with unigrams and allows at most four unigrams inside bigram components to be skipped. The computation of skip-bigram based Recall can be given as:

$$\text{R}_{\text{skip2}} = \frac{\text{Skip2}(X,Y)}{\text{C}(m,2)}, \quad (14)$$

where, Skip2(X, Y) is the number of skip-bigram matches between candidate summary Y and reference summary X, C is the combination function and C(m, 2) gives the total number of possible skip-bigrams in reference summary.

[9] https://github.com/summanlp/evaluation/tree/master/ROUGE-RELEASE-1.5.5.

Each of these ROUGE evaluation methods can generate three scores (Recall, Precision, and F-measure). Precision and F-measure scores are useful when there is no constraint on the length of the candidate summary whereas in DUC the target summary length is enforced to 100 words. As we have similar conclusions in terms of any of the three scores, for simplicity, in this paper, we only report the average recall scores generated by ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4 to compare our method with other systems.

4.3. Experimental setup

In our proposed approach to generate sentence embeddings, we utilized skip-thoughts approach (Section 3.2) proposed in Kiros et al. (2015) [15] for which the trained model was also made publicly available by the authors. We used this pre-trained skip-thoughts model after fine-tuning on CNN and DailyMail datasets (Section 4.1.1). We kept the embedding size equals to 2400, the same value as proposed in Kiros et al. (2015) since with such size of embeddings, authors have reported state-of-the-art accuracies on several NLP tasks such as semantic relatedness, paraphrase detection, image-sentence ranking, or classification. In addition, since the pre-trained skip-thought model is available only with 2400 dimensional embeddings, to generate a different size of embeddings, it would be necessary to re-train this skip-thoughts network from scratch. After re-training from scratch, it would also be required to evaluate this new size embedding on some NLP tasks to verify if the results are correct, before applying it for summarization.

The training of our auto-encoder network is also carried out on CNN and DailyMail datasets, and we evaluated our approach on DUC 2002 and TIDSumm datasets for single-document extractive text summarization. For comparative evaluation, we implemented some baseline text summarization methods for which source codes are made publicly available by their respective authors. However, in the case of evaluation on the DUC 2002 benchmark, we also took direct results from publications that use the standard experimental protocol of DUC 2002. For TIDSumm dataset, we could not compare our approach against recent state-of-the-art systems proposed in the literature as their source codes are not available publicly and reproducing them was not possible. During all our experiments, the system summary is created by arranging qualified sentences in the order of their occurrence in the input document (i.e. option B, Eq. (11)) and length of output summary is kept approximately 100 words for DUC and TIDsumm evaluation datasets. The summary length is restricted to seven sentences for Blog Summarization dataset. The training and evaluations are performed on a machine with NVIDIA GeForce GTX TITAN X GPU device with 12 GB RAM.

The final sentence score $\text{score}^{\text{f}}(\mathcal{D}, \mathcal{S}_{\text{i}})$ is determined through weighted score fusion of three parameters; $\text{score}^{ContR}(\mathcal{D}, \mathcal{S}_{\text{i}})$, $\text{score}^{Nov}(\mathcal{D}, \mathcal{S}_{\text{i}})$, and $\text{score}^{PosR}(\mathcal{D}, \mathcal{S}_{\text{i}})$ as given in Eq. (7). Our objective is the determination of optimal values of weights, i.e. α, β and γ, in order to optimize the sentence ranking. We used an unsupervised approach to learn optimal values of α, β, and γ, and they are learnt on a held out set rather than directly on the actual test dataset. In the experimentation, we first

Table 2
Auto-encoder (AE-Net) architectures showing the number of neurons in each layer.

Layer	AE-Net1	AE-Net2	AE-Net3	AE-Net4	AE-Net5	AE-Net6	AE-Net7
Input	24,000	24,000	16,800	12,000	12,000	12,000	12,000
Hidden Layer1	4096	4096	4096	2048	2048	4096	2048
Hidden Layer2	1024	1024	1024	1024	512	2048	512
Hidden Layer3	512	256	512	512	256	512	–
Hidden Layer2	1024	1024	1024	1024	512	2048	512
Hidden Layer1	4096	4096	4096	2048	2048	4096	512
Output	24,000	24,000	16,800	12,000	12,000	12,000	12,000

created a smaller held-out set of TIDSumm dataset comprising of 25 articles, i.e. different from TIDSumm in Section 4.1.3, with one ground truth summary each. We performed a grid search with combinations of $\alpha,\ \beta, \gamma \in [0,\ 0.05,\ 0.10, .., 1]$, and $\alpha+\beta+\gamma=1$ which gave us a total of 228 feasible combinations. We computed recall of ROUGE-1 and ROUGE-2 (Eq. (12)) for each combination on this set of 25 articles. We found, $\alpha=0.45,\ \beta=0.35,\ and\ \gamma=0.20$ to provide the best average over ROUGE-1 and ROUGE-2 recall on this set and thus selected them as final weight coefficients. The computation of $\alpha,\ \beta,\ and\ \gamma$ weight coefficients can be given as follows:

$$\{\alpha,\beta,\gamma\} = \underset{\alpha,\beta,\gamma\ \in\ [0,\ 0.05\ ..\ .\ 1]}{\text{argmax}} ((\text{ROUGE}{-}1\ \text{recall} + \text{ROUGE}{-}2\ \text{recall})/2) \quad (15)$$

4.4. Auto-encoder network architecture and training

The architecture of auto-encoder can be designed considering different variations including the number of hidden layers, and numbers of neurons in hidden layers. To study the impact on performance, we experimented with various configurations of the auto-encoder network by changing the number of hidden layers and the number of neurons in hidden layers. We also evaluated the system by varying the number of input sentences or the length of textual-units given as an input to the auto-encoder. In total, we tested with three different sizes (length) of textual-unit created with five, seven and ten sentences ($M=\{5,7,10\}$, e.g. $\vec{\mathcal{T}}^{10} \in \mathbb{R}^{10x2400}$ for $M=10$). Some of the best performing auto-encoder configurations (termed in this paper as AE-Net) for our text summarization task are given in Table 2. The AE-Net1 and AE-Net2 contains 24,000 neurons in the input layer pertaining to textual-units of 10 input sentences from the document, and they have a varying number of neurons in the representation layers: 512 and 256 respectively. The AE-Net3 is designed for testing the auto-encoder with input layer consisting of 16,800 neurons, which takes textual-unit of seven sentences from the document as input and has an output layer of size 512.

The AE-Net4, AE-Net5, AE-Net6 and, AE-Net7 are designed for testing the auto-encoder with textual-units of five sentences as input with 12,000 neurons in the input layer. The network configuration differs in the number of neurons as well as in the number of hidden layers such as AE-Net7 contains only two hidden layers. The network is trained for approximately 70,000 epochs using the training data (Section 4), with a mini-batch of size 128 and stochastic gradient descent for optimization with squared error loss. During training, we reduced the learning rate three times by a factor of 10 when the error stopped reducing. Finally, we terminated the training when the error stopped reducing for 10 consecutive epochs even after changing the learning rate.

4.5. Auto-encoder experiments

For the different configurations of the auto-encoder network mentioned in Table 2, we report the results on the validation set (Section 4) in Table 3. From Table 3, it can be seen that AE-Net1 is comparatively the best performing auto-encoder configuration. The AE-Net2 and AE-Net6 report a small decrease in accuracy; we believe it may be due to the lesser number of neurons in the representation layer. From our experiments, it is also visible that AE-Net7 with two hidden layers is not performing as good as compared to three hidden layer networks such as AE-Net4, AE-Net5, AE-Net6. It is also noticeable that as we increase the number of input units or the length of textual-units that can be given as an input to auto-encoder, it can capture the more global information of the document leading to better results. This pattern can be seen by analyzing the results of AE-Net1, AE-Net3, and AE-Net4 as they take ten, seven and five sentences as input and the accuracy increases as we increase the length of textual units that can be given as input to the auto-encoder. We could not experiment with textual-unit lengths of more than ten sentences, due to hardware constraints.

Table 3
ROUGE scores of SummCoder approach for different auto-encoder architectures on the validation set.

Network Conf.	AE-Net1	AE-Net2	AE-Net3	AE-Net4	AE-Net5	AE-Net6	AE-Net7
ROUGE-1	**51.7**	49.8	49.6	49.4	49.3	49.5	49.0
ROUGE-2	**27.5**	28.3	28.1	28.1	27.9	28.0	27.6
ROUGE-SU4	**28.5**	28.3	28.3	27.9	27.7	28.1	27.4
ROUGE-L	**44.6**	44.1	43.2	42.9	42.8	42.9	42.6

4.6. Approaches for comparative analysis

We perform a comparative evaluation of our proposed system with seven widely used baseline approaches and seven state-of-the-art systems published in the literature on DUC 2002. In the case of TIDSumm and blog summarization dataset, we performed an empirical evaluation of our work and compared it against baseline approaches only, as the source codes for recent state-of-the-art systems proposed in the literature are not explicitly made publicly available by their respective authors.

4.6.1. Implemented baseline extractive summarization approaches

A brief description of the baseline approaches adopted for performance comparison in this paper is as follows.

Lead selects the leading sentences in the document to form a summary. It is often used as an official baseline of DUC.

Luhn (Luhn, 1958) is the very first algorithm proposed for text summarization based on statistical features such as the frequency of words and the relative positioning of each sentence in the document.

TextRank (Mihalcea & Tarau, 2004) is a graph-based technique, which represents the document as a graph of sentences, and the edges between two sentences are connected based on the similarity between them.

LexRank (Erkan & Radev, 2004) apply the concept of eigenvector centrality to the graph representation of sentences to estimate the importance of each sentence in the document.

LSA (Latent Semantic Analysis), given by Steinberger and Jezek (2004), is a topic-based approach that tries to find salient sentences in the document by applying Singular Value Decomposition (SVD) over document matrix D of size m x n, consisting of m sentences and n number of terms.

SumBasic (Nenkova & Vanderwende, 2005) is a greedy search approximation algorithm that uses a frequency-based sentence selection component with a component to re-weight the word probabilities to minimize redundancy.

KLSum (Haghighi & Vanderwende, 2009) algorithm for text summarization generates the output summary of the document based on the criteria of Kullback-Lieber (KL) divergence.

Table 4
ROUGE Scores on DUC 2002 using baseline, state-of-the-art and SummCoder approach.

	Methods	ROUGE-1	ROUGE-2	ROUGE-SU4	ROUGE-L
Baseline Methods	Lead	43.6	21.0	20.9	40.2
	DUC 2002	48.0	22.8	–	–
	Luhn	42.5	21.2	20.4	36.5
	TextRank	47.0	19.5	21.7	42.8
	LexRank	42.9	21.1	20.5	37.1
	LSA	43.0	21.3	20.9	40.0
	KLSum	38.3	16.9	18.9	32.3
	SumBasic	39.6	17.3	19.3	35.1
State-of-the-art methods	ILP	45.4	21.3	–	42.8
	Egraph+coh	47.9	23.8	24.2	-
	Tgraph + coh	48.1	24.3	23.0	–
	URANK	48.5	21.5	–	–
	NN-SE	47.4	23.0	–	–
	CoRank	**52.6**	25.8	–	43.5
	SummaRuNNer	47.4	24.0	–	14.7
Proposed	**SummCoder**	51.7	**27.5**	**28.5**	**44.6**

4.6.2. State-of-the-art extractive summarization approaches on DUC 2002

In this section, we discuss seven state-of-the-art text summarization methods that are used for comparative analysis in the DUC 2002 benchmark. The methods are selected based on state-of-the accuracy reported by them and also which are the most cited approaches in the literature (i.e. between 2010 and 2018).

Integer Linear Programming (Woodsend & Lapata, 2010) is a phrase-based summarization technique which encodes the grammaticality constraints across phrase dependencies using integer linear programming.

Egraph (Parveen et al., 2015) is the entity graph-based system, which uses a bipartite graph of sentence and entity nodes for document representation. "Egraph+Coh." is a system developed by them, which includes a coherence measure calculated by using the average outdegree of an unweighted projection graph.

Tgraph (Parveen et al., 2015) is an unsupervised graph-based text summarization algorithm that uses Latent Dirichlet Allocation (LDA) for topic modeling. "Tgraph+coh" proposed by Parveen et al. (2015), takes coherence calculated using weighted projection graph into consideration for generating a summary.

URANK (Wan, 2010) is a unified rank methodology based on graph model, which simultaneously handles single-document and multi-document summarization.

NN-SE (Cheng & Lapata, 2016) is the Neural Summarization model for sentence extraction.

SummaRuNNer (Nallapati, Zhai, & Zhau, 2017) is a recent extractive text summarization algorithm based on RNNs.

CoRank (Fang et al., 2017) is a graph-based unsupervised extractive text summarization approach which combines word-sentence relationship with the graph-based ranking model.

4.7. Comparative analysis on DUC 2002 benchmark

For the DUC 2002 benchmark, the comparative analysis of our method is performed against (i) the baseline approaches (4.6.1), (ii) directly taking the results reported in seven state-of-the-art publications. Table 4 shows the ROUGE scores of our method, baseline approaches and state-of-the-art publications for single-document text summarization on the DUC 2002 dataset. We present the results of SummCoder using AE-Net1 configuration for auto-encoder because it is the best configuration as seen in Table 3. In general, it is depicted in Table 4 that DUC 2002's official baseline is one of the best performing algorithms for DUC 2002 single document summarization among baselines. However, our method accomplished very good ROUGE scores on DUC2002 single document extractive text summarization as compared to other summarization algorithms. We achieved a ROUGE-1 score of 51.7, ROUGE-2 score of 27.5, ROUGE-SU4 score of 28.5 and ROUGE-L score of 44.6 on this dataset. Our proposed framework (SummCoder) outperforms many of the existing state-of-the-art text summarizers on DUC 2002 dataset in terms of ROUGE-1, ROUGE-2, ROGUE-SU4 and ROUGE-L scores such as ILP, graph-based approaches, Tgraph, Egraph, URANK, and even the ones, which are based on supervised learning such as SummaRuNNer and NN-SE. Our approach has not been able to surpass the ROUGE-1 score of a recently proposed CoRank approach, though our ROUGE-2 and ROUGE-L scores which are better evaluation measures than ROUGE-1 is far better than the CoRank method.

Table 5
Comparative analysis of ROUGE scores on TIDSumm dataset.

	Method	ROUGE-1	ROUGE-2	ROUGE-SU4	ROUGE-L
Baseline	Lead	51.4	38.1	36.7	42.2
	Luhn	40.5	23.8	21.2	32.4
	TextRank	46.3	29.4	31.8	39.6
	LexRank	35.3	17.4	22.9	29.7
	LSA	40.0	21.4	24.5	33.5
	KLsum	44.0	28.8	27.2	36.0
	SumBasic	38.4	19.4	22.9	28.6
Proposed	SummCoder	58.8	48.9	45.9	49.3

4.8. Comparative analysis of TIDSumm dataset

Using TIDSumm, we evaluated our approach comparing it against seven classic extractive summarization methods (Section 4.6.1). Since the source code for the state-of-the-art methods (Section 4.6.2) is not available publicly, we could not reproduce the results of these systems on this new dataset. The obtained results are illustrated in Table 5, and it is clearly seen that compared to all the baseline algorithms evaluated, our approach obtains superior ROUGE scores on TIDSumm dataset. Lead is the second-best algorithm on the dataset followed by TextRank, KLSum, LSA, and Luhn. Whereas, SumBasic and LexRank are the worst performers. Further, to facilitate the understanding of the results, in Fig. 2, we also show the distribution of ROUGE-1, ROUGE-2, ROUGE-SU4 and ROUGE-L scores obtained using SummCoder and other implemented approaches for nine randomly selected TIDSumm documents. It can be seen that SummCoder obtains better ROUGE scores on most of the documents as compared to other approaches.

4.9. Comparative analysis on blog summarization dataset

As the source code for state-of-the-art methods is not available, we computed the results of seven baseline methods (Section 4.6.1) and our SummCoder algorithm on Blog Summarization dataset. Apart from this, we selected two additional summarization systems, Com01 (Ferreira et al., 2014) and Alg 09 (Ferreira et al., 2013), which evaluated their methods on this dataset using ROUGE-1 scores. Com01 addresses the text summarization based on TF-IDF features and sentence length whereas Alg 09 generated the summaries purely based on the length of sentences. The results are given in Table 6. It can be observed that on blog summarization dataset, SummCoder reports the highest ROUGE-1 (78.0), ROUGE-2 (71.7), ROUGE-SU4 (71. 8) and ROUGE-L (72.7), followed by Com01 and Alg09, with 77.0 and 76.0 ROUGE-1 scores respectively. Regarding the baseline methods, LSA and TextRank produced the best results, followed by Luhn, LexRank, KLSum, and SumBasic in terms of ROUGE scores. The results obtained by SummCoder

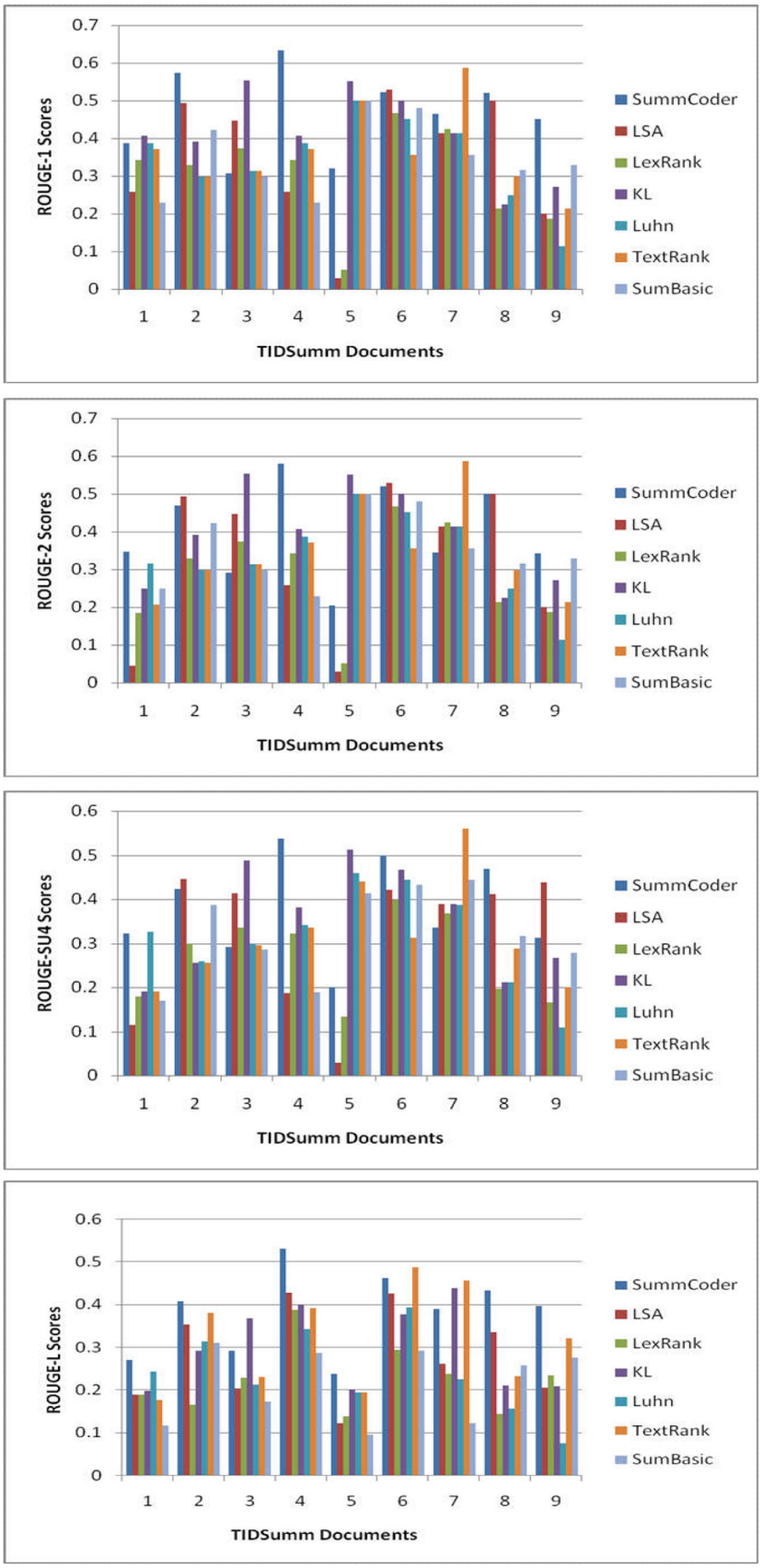

Fig. 2. Examples of ROUGE-1, ROUGE-2, ROUGE-SU4 and ROUGE-L Scores obtained by SummCoder algorithm on a subset of nine TIDSumm documents.

Table 6
Comparative analysis of ROUGE scores on Blog Summarization dataset.

	Method	ROUGE-1	ROUGE-2	ROUGE-SU4	ROUGE-L
Baseline	Lead	67.4	60.0	58.9	64.8
	Luhn	70.4	63.2	64.1	67.0
	TextRank	76.5	70.9	70.4	71.9
	LexRank	68.0	60.2	57.0	65.4
	LSA	76.9	70.9	71.4	72.0
	KLSum	63.7	55.2	55.9	60.4
	SumBasic	60.1	49.0	48.0	58.5
Proposed	**SummCoder**	**78.0**	**71.7**	**71.8**	**72.7**
Others	Com01	77.0	–	–	–
	Alg 09	76.0	–	–	–

algorithm demonstrates its robustness and its application to other domains different than news or Tor darknet domains.

4.10. Analysis of the proposed approach

In this section, we present a qualitative analysis of our proposed approach to demonstrate its effectiveness. We randomly selected one document from each DUC 2002 and TIDSumm datasets and summarized the document using our method SummCoder. The final sentence scores (Eq. (7)) obtained by each sentence in a sample DUC document along with its gold summary and the one produced with SummCoder are shown in Table 7. The score obtained for each sentence appears in brackets, at the end of each sentence. As can be seen, the reference summary and the one generated by SummCoder are quite similar. It is also noticeable that the sentences, which look more salient in the document, got much higher scores as compared to the less relevant sentences. If we compare the reference and the SummCoder summary, the first sentence in both of them is precisely the same, whereas the second sentence picked using SummCoder has the same meaning as the second sentence in the reference summary. Similarly, the third sentence in the SummCoder summary also matches with the reference one. Furthermore, in Table 8, a similar analysis is presented for a randomly selected document from TIDSumm dataset. The final scores for the sentences and the reference and SummCoder summary are shown in Table 8. It is visible that our technique can produce a good summary of the document, which matches with most of the sentences from the reference summary. Thus, we can infer from the above experiments that our method is suitable for summarizing the contents in TIDSumm dataset.

From results in Tables 4 and 5, it is seen that ROUGE scores obtained by our approach are little higher on TIDSumm dataset than DUC 2002. It should be noted that in the case of DUC 2002, a few sentences in gold summary have been reformulated slightly compared to sentences in the original source document. However, in TIDSumm, the gold summaries are created by picking whole sentences without any modification. Therefore, our method is purely extractive and selects the sentences as they are from the input document and hence performed a little better on TIDSumm. We also realized that in TIDSumm, there are a significant number of web pages where the text is not very narrative but descriptive, and there is enough number of pages with short sentences describing products or conversations, from markets or forums. For these reasons, extractive summarization works very well in this context and, as can be seen in the obtained results, and our proposal performs especially well. This can also be attributed to improved performance of SummCoder approach on TIDSumm.

Moreover, in Table 9, we also present an example from DUC 2002 dataset for which SummCoder has not performed well and generated a summary that is a little different from the gold summary. From the summaries, it can be observed that the first sentence in the document, which is assumed to convey an important message in our approach, is not present in the gold summary. Another assumption in our approach is that sentences with a high number of keywords are normally more important, and because of this the last sentence of the document has obtained a higher score (i.e. 0.90456), however in gold summary this sentence is not included. Another point, which can be highlighted, is that SummCoder always picks the whole sentence from the input document. However, some sentences in gold summary, probably with the intention to make the summary more effective, they picked the important part of one sentence and even combined it with another part picked from another sentence. For example, the second sentence in the gold summary "Called teaming and lading, Maxwell and his son Kevin began to buy and sell enormous amounts of foreign currencies." is created by taking some parts from fifth sentence (i.e. Starting about July 4, 1991, MCC's treasury, under the

Table 7
Illustration of a sample document from DUC 2002 and its reference and SummCoder summary.

Final scores obtained by each sentence of a DUC 2002 document with SummCoder approach	
The parents of a Peace Corps worker learned Thursday that their son, a volunteer in Nepal, was safe following an earthquake last weekend that killed at least 750 people in that country and neighboring India. [Score: 0.96927] "They called this morning, that's the first we had heard about anything since the quake," said Arthur Giaquinta, whose 24-year-old son, Peter, lives in a village 10 miles north of the hard-hit Dharan Bazar township. [Score: 0.88924] Jim Flanigan, a Peace Corps spokesman in Washington, said Peter Giaquinta was "fine as far as we know." [Score: 0.88716] In Boston, the brother of Peace Corps volunteer Bettyjean Bouffard, 27, said Thursday she called her relatives to say she was safe. [Score: 0.94268] The three, he said, are from Michigan, Wisconsin and the Philadelphia area; he declined to release their names. [Score 0.90563] The call, said Dave Bouffard, was the first news the family had of her safety. [Score 0.888119] "We're ecstatic," he said, adding that his sister had been living with a Nepalese family five miles from the earthquake epicenter. [Score 0.92114] There are no Peace Corps members in India, he said. [Score 0.88565] "We have a problem with roads out, monsoon rains," Flanigan said. [Score 0.88522] Flanigan said the Peace Corps has been able to track down all but three of the 157 volunteers in Nepal. [Score 0.88464] "In the best of times, reaching some of these volunteers is several days." [Score 0.86771]	
Reference Summary	SummCoder Summary
The parents of a Peace Corps worker learned Thursday that their son, a volunteer in Nepal, was safe following and earthquake last weekend that killed at least 750 people in that country and neighboring India. Another Peace Corps worker, Bettyjean Bouffard also called her relatives to say she was safe. She had been living with a Nepalese family five miles from the earthquake epicenter. The Peace Corps has been able to track down all but three of its volunteers in Nepal. Monsoon rains and washed-out roads kept them from reaching some of the volunteers. There aren't any Peace Corps volunteers in India.	The parents of a Peace Corps worker learned Thursday that their son, a volunteer in Nepal, was safe following an earthquake last weekend that killed at least 750 people in that country and neighboring India. In Boston, the brother of Peace Corps volunteer Bettyjean Bouffard, 27, said Thursday she called her relatives to say she was safe. "We're ecstatic," he said, adding that his sister had been living with a Nepalese family five miles from the earthquake epicenter. The three, he said, are from Michigan, Wisconsin and the Philadelphia area; he declined to release their names.

Table 8
Illustration of a sample document from TIDSumm and its reference and SummCoder summary.

Final scores obtained by each sentence of a TIDSumm document with SummCoder approach	
We offer 20/50/100 Euro & USD bills of good quality, Cotton, micro-print, provided with watermarks, security thread, pass pen test, etc. [Score: 0.96946] These are made with exactly the same techniques the ECB and the Federal Reserve use to make genuine currency. [Score: 0.93523] For amounts under 500 euro every euro bill costs 40% off its value. [Score: 0.94403] For amounts under 2000 euro every euro bill costs 30% off its value we can also make bigger proportion. [Score: 0.92424] Please ask us for larger quantities. [Score: 0.90206] For amounts under 2000 usd every usd bill costs 30% off its value. [Score: 0.90453] Once you have given us your order you send us your address of receivement and send the money to our bitcoin wallet. [Score: 0.90835] Once we have comfirmed your payment we ship them the same day in ceiled iphone boxes. shipping within the eu is free and takes days. [Score: 0.89171] We we want to be personal and discrete at the same time. [Score: 0.89899] So please email us to negotiate get more information or to make deal. [Score: 0.89567] countr safe mail.net rules we ship around the whole world but we only garantuee delivery within european union and the us. [Score: 0.88847] shipping is free within europe or canada and the us and takes days payments goes in bitcoins only you pay everything before we send your package. [Score: 0.89022] we do already take lot of the risk. every email you send at countr safe mail.net will be answered within hours. [Score: 0.88698] why we are not scam? [Score: 0.86716] because we want to live from our profession on the long term. [Score: 0.86938] how I it looks eur and usd counterfeits. [Score: 0.87789]	
Reference Summary	SummCoder Summary
We offer 20/50/100 Euro & USD bills of good quality, Cotton, micro-print, provided with watermarks, security thread, pass pen test, etc. These are made with exactly the same techniques the ECB and the Federal Reserve use to make genuine currency. For amounts under 500 euro, every euro bill costs 40% off its value. For amounts under 2000 euro, every euro bill costs 30% off its value. For amounts under 2000 USD, every USD bill costs 30% off its value. We can also make bigger proportion. Please ask us for larger quantities.	We offer 20/50/100 euro and usd bills of good quality, cotton, micro-print, provided with watermarks, security thread, pass pen test, etc. These are made with exactly the same techniques the ECB and the Federal Reserve use to make genuine currency. For amounts under 500 euro, every euro bill costs 40% off its value. For amounts under 2000 euro, every euro bill costs 0% off its value. We can also make bigger proportion. once you have given us your order you send us your address of receivement and send the money to our bitcoin wallet.

Table 9
Illustration of a sample document from DUC 2002 and its summaries on which SummCoder did not obtain good ROUGE.

The final score obtained by each sentence of a DUC 2002 document	
About 20 banks became con-cerned last summer at the behaviour of businesses run by Robert Maxwell several months before he died and his business empire collapsed. score 0.96451 Their concern arose when Maxwell Communication Corporation made late payments to them, a Financial Times investigation has discovered. score 0.89173 Yet although some banks became alarmed at the implications for the financial health of his empire, they did not discuss their concerns with each other or with authorities such as the Bank of England. score 0.82220 Mr Basil Brookes, MCC's former finance director, has described his shock to the FT at discovering last August that MCC had failed to settle foreign currency transactions with about 20 banks on time. score 0.82216 Starting about July 4, 1991, MCC's treasury, under the authority of Maxwell and his son, Kevin, had begun to buy and sell enormous amounts of foreign currencies. score 0.85440 In these 'spot' transactions, MCC would buy a currency in one financial centre promising to make payment in another centre. score 0.84337 On a number of occasions, MCC received the currency, but did not make the payment - thereby giving itself an unauthorised loan. score 0.82230 Mr Brookes says about 20 banks, including Lloyds, Bank of America, Goldman Sachs and Swiss Bank Corporation, were involved in the flurry of foreign exchange deals. score 0.82315 At least Dollars 100 m (Pounds 55 m) in unapproved credit is thought to have been raised this way. score 0.82255 A former Maxwell employee said some appeared to have been channelled into private Maxwell businesses. score 0.82403 One banker said attempts to raise money by making late payments in the foreign market was a relatively well-known ruse with the nickname called 'teaming and lading'. score 0.82207 It also emerged yesterday that National Westminster Bank, the biggest lender to the Maxwell private companies, felt badly let down by Maxwell because he failed to deliver funds on time at the end of July. score 0.82242 At a meeting between Maxwell and Mr John Melbourn, a NatWest director, on July 10, NatWest refused Maxwell's request for a loan of Pounds 45 m. score 0.82385 NatWest was upset three weeks later when it provided funds - less than Pounds 100 m - to the Maxwell empire on the understanding that it would be repaid the same day, and then found that repayment was not made. score 0.82399 NatWest was forced to convert this 'intra-day' transaction into a short-term loan, which was eventually repaid. score 0.82224 The bank told both Robert Maxwell and Mr Kevin Maxwell that it was concerned that repayment had not been made, but it did not discuss the problem with other banks or with the Bank of England. score 0.82384 Instead, it pressed on with its policy of reducing the size of its loans to the Maxwell businesses. score 0.82234 NatWest's loans to the Maxwell empire now total Pounds 182 m, half the level of two years ago. score 0.86477 Later last year, Robert Maxwell made many repayments to banks by looting pension fund assets and also carrying out unusual fundraising exercises with US investment banks, from which tens of millions of dollars were borrowed at a high price. score 0.90456	
Reference Summary	SummCoder summary
In an effort to raise funds, Maxwell Communications Corporation made late payments on the foreign market. Called teaming and lading, Maxwell and his son Kevin began to buy and sell enormous amounts of foreign currencies. Maxwell would buy currency in one financial center and promise to repay it in another. MCC received the currency, but on a number of occasions, did not make the payment, giving itself an unauthorized loan. About 20 banks, including Lloyds, Bank of America, Goldman Sachs, and Swiss Bank Corporation were involved in the exchange deals. At least 100 m dollars in credit is thought to have been raised this way.	About 20 banks became con-cerned last summer at the behaviour of businesses run by Robert Maxwell several months before he died and his business empire collapsed. Their concern arose when Maxwell Communication Corporation made late payments to them, a Financial Times investigation has discovered. NatWest's loans to the Maxwell empire now total Pounds 182 m, half the level of two years ago. Later last year, Robert Maxwell made many repayments to banks by looting pension fund assets and also carrying out unusual fundraising exercises with US investment banks, from which tens of millions of dollars were borrowed at a high price.

authority of Maxwell and his son, Kevin, had begun to buy and sell enormous amounts of foreign currencies.) and eleventh sentence (i.e. One banker said attempts to raise money by making late payments in the foreign market was a relatively well-known ruse with the nickname called 'teaming and lading'). Although our method is not so intelligent, in the future we would like to improve summarization by considering phrases as a smaller unit than a sentence as taken in current work, looking this way to improve the summarization.

5. Conclusion and future work

In this work, we have presented a novel unsupervised framework named as SummCoder for single-document extractive text summarization using deep auto-encoders and sentence embeddings. The proposed approach creates document extracts by ranking and selecting sentences from the input text. To rank the sentences, we proposed three metrics: sentence content relevance, sentence novelty, and sentence position jointly with a fusion scheme that allows generating the final sentence selection score. To compute the sentence content relevance metric, we introduced an auto-encoder based method, which exploits sentence vector representations, obtained using skip-thought vectors. By inheriting the abstractive ability of auto-encoders, our approach learns to discriminate low dimensional latent document representations extracted from the encoder part of the trained auto-encoder model. The insight underlying the use of auto-encoder based latent representations is that the low dimensional document representation captures all the key concepts of the document while minimizing redundancy. We utilized this latent representation to compute the sentence content relevance score to determine the saliency of each sentence in the document.

Furthermore, in our framework, the sentence novelty score is obtained by exploiting the similarity between sentences represented in a learned embedding space. If two sentences are close to each other in the embedding space they are considered carrying more similar or redundant information than that when they are far away. Thus, we incorporate the sentence novelty score to minimize the redundancy in the summary document. The third metric, sentence position score is a hand-crafted feature which dynamically gives more weight to the sentences placed at the beginning of the document based on the document length. The summary is generated by ranking sentences as per the weighted fusion of scores obtained through our proposed sentence relevance, position and novelty metrics. Our approach demonstrates the effectiveness of using deep learning techniques for the conceptual representation of the document by exploiting sentence embeddings. Furthermore, a new text summarization benchmark dataset named Tor Illegal Documents Summarization (TIDSumm) is also introduced. The TIDSumm contains manually created two sets of ground truth summaries for 100 documents extracted from onion websites in Tor darknet, which host content related to illegal activities apart from legal ones. This kind of text summaries will be useful for Law Enforcement Agencies to track or to monitor suspicious activities carried across the Tor network.

We employed an extensive experimental analysis to ascertain the validity of the proposed text summarization method. For comparative analysis, we utilized DUC 2002, TIDSumm and Blog Summarization datasets and seven well-known baseline approaches from literature. The comparative study is conducted using well-known Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metric for text summarization. For the DUC 2002 benchmark, we also presented a comparative analysis of our system against seven recent state-of-the-art methods. Our system obtained highly competitive performance on the DUC 2002 dataset, and we found that it outperforms most of the recent state-of-the-art methods, even those based on supervised learning. Therefore, we obtained empirical evidence that our approach is quite robust and also performed well on TIDSumm and Blog Summarization datasets which are from a different domain than news. Through a series of experiments, we also demonstrated the impact of different configurations of the auto-encoder network on the performance of our system. The overall approach is unsupervised and does not require gold summaries to train the text summarization system. The impressive accuracy on DUC 2002, TIDSumm and Blog Summarization datasets, which are from different domains, also suggests that the proposed framework generalizes very well on new samples or domains.

In the future, we would explore additional variations of auto-encoder networks for summarization task. We also plan to extend our approach for query-based text summarization and multi-document text summarization.

Credit authorship contribution statement

Akanksha Joshi: Investigation, Conceptualization, Methodology, Software, Writing - original draft, Data curation. **E. Fidalgo:** Project administration, Data curation, Investigation, Validation, Resources. **E. Alegre:** Supervision, Writing - review & editing, Funding acquisition. **Laura Fernández-Robles:** Validation, Writing - review & editing.

Acknowledgements

This research was supported by the INCIBE grant "INCIBEI-2015-27359" corresponding to the "Ayudas para la Excelencia de los Equipos de Investigacion avanzada en ciberseguridad" and also by the framework agreement between the University of Leon and INCIBE (Spanish National Cybersecurity Institute) under Addendum 22 and 01. We acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

Abuobieda, A., Salim, N., Albaham, A. T., Osman, A. H., & Kumar, Y. J. (2012). Text summarization features selection method using pseudo genetic-based model. In *Proceedings of 2012 international conference on information retrieval and knowledge management, CAMP'12* (pp. 193–197).

Alami, N., En-nahnahi, N., Ouatik, S. A., & Meknassi, M. (2018). Using unsupervised deep learning for automatic summarization of Arabic documents. *Arabian Journal for Science and Engineering, 43*, 7803–7815.

Alguliev, R. M., & Aliguliyev, R. M. (2008). Automatic text documents summarization through sentences clustering. *Journal of Automation and Information Sciences, 40*(9), 53–63.

Al-Nabki, M., Fidalgo, E., Alegre, E., & De Paz, I. (2017). Classifying Illegal Activities on Tor Network Based on Web Textual Contents. In *Proceedings of European Chapter of the Association for Computational Linguistics* (pp. 35–43).

Bordes, A., Chopra, S., & Weston, J. (2014). Question answering with subgraph embeddings. In *Proceedings of empirical methods in natural language processing* (pp. 615–620).

Cao, Z., Chen, C., Li, W., Li, S., Wei, F., & Zhou, M. (2016). Tgsum: Build tweet guided multi-document summarization dataset. In *Proceedings of thirtieth AAAI conference on artificial intelligence* (pp. 2906–2912).

Cao, Z., Wei, F., Dong, L., Li, S., & Zhou, M. (2015). Ranking with recursive neural networks and its application to multi-document summarization. In *Proceedings of the twenty-ninth AAAI conference on artificial intelligence* (pp. 2153–2159).

Cao, Z., Wei, F., Li, S., Li, W., Zhou, M., et al. (2015). Learning summary prior representation for extractive summarization. In *Proceedings of association for computational linguists* (pp. 829–833).

Cao, Z., Li, W., Li, S., Wei, F., & Li, Y. (2016). Attsum: Joint learning of focusing and summarization with neural attention. In *Proceedings of COLING 2016, the 26th international conference on computational linguistics* (pp. 547–556).

Carbonell, J., & Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval - SIGIR '98* (pp. 335–336).

Ceylan, H., Mihalcea, R., Öyertem, U., Lloret, E., & Palomar, M. (2010). Quantifying the Limits and Success of Extractive Summarization Systems Across Domains. In *Proceedings of Human language technologies: The 2010 annual conference of the North American Chapter of the ACL (NACLO 2010)* (pp. 903–911).

Cheng, J., & Lapata, M. (2016). Neural summarization by extracting sentences and words. In *Proceedings of 54th annual meeting of the association for computational linguistics* (pp. 484–494).

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research, 12*, 2493–2537.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Proceedings of deep learning and representation learning workshop: NIPS 2014* (pp. 1–9).

Denil, M., Demiraj, A., & Freitas, N. D. (2015). Extraction of Salient Sentences from Labelled Documents. *CoRR*. arXiv: 1412.6815.

Dunlavy, D. M., O'Leary, D. P., Conroy, J. M., & Schlesinger, J. D. (2007). QCS: A system for querying, clustering and summarizing documents. *Information Processing and Management, 43*(6), 1588–1605.

Edmundson, H. P. (1969). New methods in automatic extracting. *Journal of the Association for Computing Machinery, 16*(2), 264–285.

Erkan, G., & Radev, D. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research, 22*(1), 457–479.

Fang, C., Mu, D., Deng, Z., & Wu, Z. (2017). Word-sentence co-ranking for automatic extractive text summarization. *Expert Systems with Applications, 72*, 189–195.

Fattah, M. A., & Ren, F. (2009). GA, MR, FFNN, PNN and GMM based models for automatic text summarization. *Computer Speech and Language, 23*(1), 126–144.

Ferreira, R., Cabral, L., Lins, D., Pereira, G., Freitas, F., Cavalcanti, G., et al. (2013). Assessing sentence scoring techniques for extractive text summarization. *Expert Systems with Applications, 40*, 5755–5764.

Ferreira, R., Freitas, F., Cabral, L., Lins, R., Lima, R., Franca, G., et al. (2014). A context based text summarization system. In *Proceedings of 11th IAPR international workshop on document analysis systems* (pp. 66–70).

Galgani, F., Compton, P., & Hoffmann, A. (2012). Citation based summarisation of legal texts. In *Proceedings of PRICAI 2012: trends in artificial intelligence* (pp. 40–52).

Gambhir, M., & Gupta, V. (2017). Recent automatic text summarization techniques: A survey. *Artificial Intelligence Review, 47*(1), 1–66.

Genest, P., Gotti, F., & Bengio, Y. (2011). Deep learning for automatic summary scoring. In *Proceedings of the workshop on automatic text summarization* (pp. 17–28).

Gong, Y., & Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 19–25).

Gupta, P., Pendluri, V. S., & Vats, I. (2011). Summarizing text by ranking text units according to shallow linguistic features. In *Proceedings of the 13th international conference on advanced communication technology* (pp. 1620–1625).

Haghighi, A., & Vanderwende, L. (2009). Exploring content models for multi-document summarization. In *Proceedings of human language technologies: The 2009 annual conference of the north American chapter of the association for computational linguistics on - NAACL '09* (pp. 362–370).

Hermann, K. M., Kočiský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., et al. (2015). Teaching machines to read and comprehend. In *Proceedings of the 28th international conference on neural information processing systems* (pp. 1693–1701).

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science, 313*(5786), 504–507.

Hochreiter, S., & Urgen Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.

Hu, M., Sun, A., & Lim, E.-P. (2007). Comments-oriented blog summarization by sentence extraction. In *Proceedings of the 16th ACM conference on information and knowledge management* (pp. 901–904).

Hu, M., Sun, A., & Lim, E.-P. (2008). Comments-oriented document summarization: Understanding documents with readers' feedback. In *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 291–298).

Huang, L., He, Y., Wei, F., & Li, W. (2010). Modeling document summarization as multi-objective optimization. In *Proceedings of the third international symposium on intelligent information technology and security informatics* (pp. 382–386).

Jean, S., Cho, K., Memisevic, R., & Bengio, Y. (2015). On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing* (pp. 1–10).

Kageback, M., Mogren, O., Tahmasebi, N., & Dubhashi, D. (2014). Extractive summarization using continuous vector space models. In *Proceedings of the 2nd workshop on continuous vector space models and their compositionality (CVSC)* (pp. 31–39).

Kaikhah, K. (2004). Text summarization using neural networks. *WSEAS Transactions on Systems, 3*(2), 960–963.

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. In *Proceedings of association for computational linguists* (pp. 655–665).

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of empirical methods in natural language processing* (pp. 1746–1751).

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. In *Proceedings of international conference on learning representations* (pp. 1–14).

Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R. S., Torralba, A., Urtasun, R., et al. (2015). Skip-thought vectors, (786). In *Proceedings of neural information processing systems* (pp. 1–11).

Kobayashi, H., Yatsuka, M., & Taichi, N. (2015). Summarization based on embedding distributions. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 1984–1989). (September).

Ko, Y., & Seo, J. (2008). An effective sentence-extraction technique using contextual information and statistical approaches for text summarization. *Pattern Recognition Letters, 29*(9), 1366–1371.

Koupaee, M., & Wang, W. Y. (2018). WikiHow: A large scale text summarization dataset. *CoRR*. arXiv: 1810.09305.

Kumar, G., & D'Haro, L. F. (2015). Deep autoencoder topic model for short texts. *Procedings of IWES: international workshop on embeddings and semantics, held in conjunction with the SEPLN 2015 Spanish Society for Natural Language Processing*: 1.

Le, Q. V., & Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of international conference on machine learning* (pp. 1188–1196).

Li, L., Zhou, K., Xue, G.-R., Zha, H., & Yu, Y. (2009). Enhancing diversity, coverage and balance for summarization through structure learning. In *Proceedings of the 18th international conference on world wide web* (pp. 71–80).

Li, P., Wang, Z., Lam, W., Zhaochun Ren, Z., & Bing, L. (2017). Salience estimation via variational auto-encoders for multi-document summarization. In *Proceedings of international conference on artificial intelligence* (pp. 3497–3503).

Lin, C. Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Proceedings of the workshop on text summarization branches out (WAS 2004)* (pp. 25–26).

Lin, H., & Bilmes, J. (2011). A class of submodular functions for document summarization. *Computational Linguistics, 1*, 510–520.

Liu, P., Saleh, M., Pot, E., Goodrich, R., Sepassi, R., Kaiser, L., & Shazeer, N. (2018). Generating Wikipedia by Summarizing Long Sequences. In *Proceedings of International Conference on Learning Representations, ICLR*.

Lloret, E., & Palomar, M. (2013). Tackling redundancy in text summarization through different levels of language analysis. *Computer Standard & Interfaces, 35*(5), 507–518.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research Development, 2*(2), 159–165.

Makhzani, A., & Frey, B. J. (2014). k-Sparse Autoencoders. *CoRR*. arXiv: 1312.5663.

Mcdonald, R. (2007). A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European conference on IR research* (pp. 557–564).

Mehta, P., Arora, G., & Majumder, P. (2018). Attention based sentence extraction from scientific articles using pseudo-labeled data. *CoRR*. arXiv: 1802.04675.

Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing order into texts. In *Proceedings of empirical methods on natural language processing* (pp. 404–411).

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In *Proceedings of international conference on learning representations* (pp. 1–12).

Nallapati, R., Zhai, F., & Zhou, B. (2017). SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the 31st AAAI conference* (pp. 3075–3081).

Nallapati, R., Zhou, B., & Ma, M. (2018). Classify or select: Neural architectures for extractive document summarization. *CoRR*. arXiv: 1611.04244.

Nallapati, R., Zhou, B., Santos, C. N. dos, Gulcehre, C., & Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of the SIGNLL conference on computational natural language learning* (pp. 280–290).

Narayan, S., Cohen, S. B., & Lapata, M. (2018). Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 16th annual conference of the North American chapter of the association for computational linguistics: human language technologies (NAACL-HLT)* (pp. 1747–1759).

Nenkova, A., & Vanderwende, L. (2005). The impact of frequency on summarization. *Technical report, Microsoft Research* (MSR-TR-2005-101).

Ouyang, Y., Li, W., Li, S., & Lu, Q. (2011). Applying regression models to query-focused multi-document summarization. *Information Processing and Management, 47*(2), 227–237.

Palangi, H., Deng, L., Shen, Y., Gao, J., He, X., Chen, J., et al. (2016). Deep sentence embedding using long short term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing, 24*(4), 694–707.

Pagliardini, M., Gupta, P., & Jaggi, M. (2017). Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of NAACL 2018 - conference of the North American chapter of the association for computational linguistics* (pp. 528–540).

Parveen, D., Ramsl, H.-M., & Strube, M. (2015). Topical coherence for graph-based extractive summarization. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 1949–1954).

Rifai, S., Vincent, P., Muller, X., Glorot, X., & Bengio, Y. (2011). Contractive auto-encoders: explicit invariance during feature extraction. In *Proceedings of international conference on machine learning* (pp. 833–840).

Rush, A. M., Chopra, S., & Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *Proceedings of empirical methods on natural language processing* (pp. 379–389).

Santos, C. N. dos, & Gatti, M. (2014). Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings the 25th international conference on computational linguistics: technical papers* (pp. 69–78).

Sarkar, K. (2010). Syntactic trimming of extracted sentences for improving extractive multi-document summarization. *Journal of Computing, 2*, 177–184.

Sinha, A., Yadav, A., & Gahlot, A. (2018). Extractive text summarization using neural networks. *CoRR*. arXiv: 1802.10137.

Socher, R., Perelygin, A., & Wu, J. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of conference on empirical methods in natural language processing* (pp. 1631–1642).

Steinberger, J., & Jezek, K. (2004). Using latent semantic analysis in text summarization and summary evaluation. In *Proceedings of 7th International Conference on Information System Implementation and Modeling* (pp. 93–100).

Svore, K. M., Way, M., Vanderwende, L., & Burges, C. J. C. (2007). Enhancing single-document summarization by combining RankNet and third-party sources. *Computational Linguistics*, 448–457.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Antoine Manzagol, P. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 3371–3408.

Wan, X. (2008). Using only cross-document relationships for both generic and topic-focused multi-document summarizations. *Information Retrieval, 11*, 25–49.

Wan, X. (2010). Towards a unified approach to simultaneous single-document and multi-document summarizations. *In Proceedings of the 23rd international conference on computational linguistics (Coling)*, 1137–1145.

Wang, D., Li, T., Zhu, S., & Ding, C. (2008). Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 307–314).

Wood, J. (2010). The darknet: A digital copyright revolution (PDF). *Richmond Journal of Law and Technology, 16*(4), 15–17.

Woodsend, K., & Lapata, M. (2010). Automatic generation of story highlights. In *Proceedings of the 48th annual meeting of the association for computational linguistics* (pp. 565–574).

Wu, Y., & Hu, B. (2018). Learning to extract coherent summary via deep reinforcement learning. In *Proceedings of thirty-second AAAI conference on artificial intelligence (AAAI-18)* (pp. 5602–5609).

Yasunaga, M., Zhang, R., Meelu, K., Pareek, A., Srinivasan, K., & Radev, D. (2017). Graph-based neural multi-document summarization. In *Proceedings of the 21st conference on computational natural language learning (CoNLL 2017)* (pp. 452–462).

Yeh, J. Y., Ke, H. R., Yang, W. P., & Meng, I. H. (2005). Text summarization using a trainable summarizer and latent semantic analysis. *Information Processing and Management, 41*(1), 75–95.

Yin, W., & Pei, Y. (2015). Optimizing sentence modeling and selection for document summarization. In *Proceedings of international joint conference on artificial intelligence* (pp. 1383–1389).

Yousefi-Azar, M., & Hamey, L. (2017). Text summarization using unsupervised deep learning. *Expert Systems with Applications, 68*, 93–105.

Yogatama, D., & Smith, N. A. (2015). Extractive summarization by maximizing semantic volume. In *Proceedings of empirical methods in natural language processing* (pp. 1961–1966).

Zajic, D., Dorr, B., & Jimmy, L. (2008). Single-document and multi-document summarization techniques for email threads using sentence compression. *Information Processing & Management, 44*, 1600–1610.

Zhang, C., Sah, S., Nguyen, T., Peri, D., Loui, A., Salvaggio, C., et al. (2017). Semantic sentence embeddings for paraphrasing and text summarization. In *Proceedings of IEEE global conference on signal and information processing* (pp. 705–709).

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Proceedings of the advances in neural information processing systems* (pp. 649–657).

Zhong, S. H., Liu, Y., Li, B., & Long, J. (2015). Query-oriented unsupervised multi-document summarization via deep learning model. *Expert Systems with Applications, 42*(21), 8146–8155.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., et al. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision* (pp. 19–27).