Query-oriented Single-document Summarization Using Unsupervised Deep Learning

by

Mahmood Yousefiazar

A Thesis Presented in Partial Fulfillment
Of the Requirements for the Degree
Master of Research

Department of Computing
Macquarie University
October 2015

The Statement of Originality

This is to certify that the work in this thesis has not been submitted for a higher degree to any other university or institution.

محمود یوسفی آذر

Mahmood Yousefiazar

Abstract

Over the past half a century, machine-based text summarization has been addressed from many different perspectives in a variety of application domains. Deep neural networks recently show promising results for text summarization and this thesis explores this application domain. In this research study, a deep auto-encoder is used to rank sentences based on the most salient information. More precisely, a deep neural network has been used for extractive query-oriented single-document summarization. Also, the use of an Ensemble Noisy Auto-Encoder (ENAE) for this task has been evaluated. ENAE is a stochastic version of an auto-encoder that adds noise to the input text and selects the top sentences from an ensemble of runs. Our experiments show that although a deep auto-encoder can be an effective summarizer, deep auto-encoders trained with stochastic noise in the input and run multiple times with different noise in the input can make improvements. The architecture of ENAE changes the application of the auto-encoder from a deterministic feed-forward network to a stochastic model. To cover a wide range of topics and structures, we perform experiments on two different publicly available email corpora that are specifically designed for text summarization.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I would like to express my deepest gratitude to Associate Prof Mark Dras and Dr. Len Hamey for their insightful comments and for being supportive supervisors throughout my research at Macquarie University. Their expertise and advice have been very helpful in my thesis.

My sincere thanks also goes to Dr. Christophe Doche, Prof. Tracy Rushmer and the rest of my thesis committee for their encouragement and moral support.

I would also like to thank my parents, Mohammad and Sara, and my sisters, Samira and Sima, for supporting me spiritually throughout writing this thesis and my life in general.

Without the support, encouragement, and dedication to assist me, this dissertation would not have been possible.

This research was funded under an iMQRES Scholarship in Macquarie University.

Chapter 1

1. Introduction

The main goal of text summarization is to produce a condensed version of the original texts using an automatic technique. Indeed, providing a user with a summary of a document can be highly informative and greatly facilitates finding the required information in a large volume of the structured and unstructured data. Indeed, summarization and text search can be considered as the two pieces of the information extraction puzzle that complement each other. Having a returned set of documents from a text search engine, text summarizers generate a summary of a document for a quick examination. This is what most popular search engines (e.g. Google) are doing when they present the search results, including snippets of text that are related to the query. Text summarization can play an important role in different application domains. For instance, when performing a search in the mailbox according to a keyword, the user could be shown short summaries of the relevant emails. This is especially attractive when using a smart phone with a small screen.

With the advances of information technology and the increase of digital data available world-wide it is becoming impossible to summarize information manually. Not only the amount of data is challenging but summarization, especially for technical texts, requires a considerable number of qualified unbiased experts, considerable time and budget. Even with experts, the quality of the abstract of a text might vary widely since human summarization is subjective. The drawbacks of manual text summarization grabbed experts' attention to propose a machine based approach, from the late 50's (Luhn 1958).

Generally speaking, summarization can be categorized into two distinct classes: abstractive and extractive (Hovy 2005). In the abstractive summarization the summarizer has to re-generate the extracted content; however, in extractive category the sentences have to be ranked based on the most salient information. Extractive summarization has become a more popular research field mainly because truly abstractive summarization requires a complex linguistic process and real-word knowledge. In many research studies extractive summarization is equally known as sentence ranking (Edmundson 1969, Carbonell and Goldstein 1998, Maybury 1999). Both extractive and abstractive summarization systems differ according to function, target reader, processing level point of view, and applied tools.

From another perspective, a summarizer will be providing a generic summary or query-based summary (Mani 2001). A generic summary provides a general sense of the document's contents whereas a query-based summary presents the contents of the document that are closely related to the query. A query-based summarizer aims to retrieve and summarize a document or a set of documents satisfying a request for information expressed by a user query. The query-based summarization is sometimes also called "user-focused summarization".

In practice, most researchers in automated text summarization have identified three distinct stages: topic identification, topic fusion and summary generation (Hovy 2005). Figure 2-1 shows the three stages. The first stage is where the system identifies the most important unit(s) (words, sentences, paragraphs, etc.) and can either list them (in automatic extracting) or display them diagrammatically (having a schematic summary). In the topic fusion stage (or interpretation stage), in which extractive summarization systems from abstractive systems can be distinguished, identified topics are fused and expressed by a new formulation (i.e. concepts and words) that is not presented in the original text. In the third phase, with having the summary content through abstracting or extracting information, in internal notation, it is required to use natural language generation techniques to produce the output summary (Hovy 2005). Most approaches only embody the first stage (topic identification) as a text summarization system.

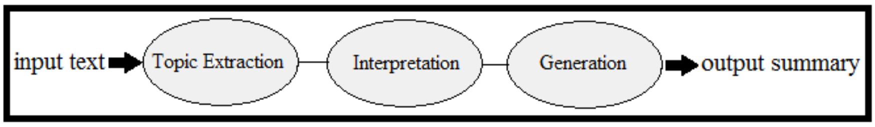


Figure 1-1 Text summarization based on the topic identification + interpretation + generation.

When it comes to applying a technique, defining a summarizer's task is a key factor. For example, the summarizer will work on either single document or multi-document and the purpose of summarizer can be either generic or query-based summarization. The target reader, for whom summarization will be on the basis of domain specific (e.g. medicine) or not, is quite decisive as well.

There have been investigations on how empirical methods have been applied to develop a text summarizer (Nenkova 2005, Das and Martins 2007). Evaluation measures such as coverage (i.e. to show the performance of a system in capturing content selection), readability and coherence, in both single and multi-document summarization, have been tested and compared. The evidence shows that to have the best performing system with reliable statistical significance the function of summarizer is quite significant.

In addition to statistical, linguistic and graph-based methods that can be used separately and in combination, most recent research on document summarization uses machine learning and artificial intelligence techniques. To be more exact, closely related fields (e.g. information retrieval and text mining) have been investigated and adapted for text summarization. The techniques vary based on the application and the objective of the summarization.

1.1 Automatic Text Summarization Using Unsupervised Deep Learning

One of the most recent and successful methods for text processing is deep learning (LeCun et al. 2015, Schmidhuber 2015). In general, most machine learning techniques for Natural Language Processing (NLP) are limited to numerical optimization of weights for human designed features from the text; one of the key characteristics of deep or representation learning, however, is to automatically develop features or representations from the original text, appropriate for NLP tasks. This characteristic and other capabilities have motivated many researchers to explore the application of deep structures in NLP, for instance in text summarization. Indeed, in order to learn complicated functions that can present high-level abstractions, such as NLP tasks, a deep architecture may be required (Deng and Yu 2014).

Given a text, our system makes a summary based on either heading of the text or key phrases. Indeed, the model was designed to rank and extract sentences that are semantically similar to the subject of a text or previously selected key phrases of a text. The pivot of the model is a deep auto-encoder (AE) as an unsupervised model and the key factor of the model is the word representation (see section 3.3). In this thesis, AE was used to map the input (i.e. word representation) to a semantic space where meaningfully similar sentences can have less distance from each other compared to other sentences. Also, a voting algorithm, in which sentences can be selected, has been explored. The architecture of this ensemble noisy auto-encoder (ANAE) changes the application of the AE from a deterministic feed-forward network to a stochastic model.

To evaluate the model, a series of experiments have been conducted on two different publicly available email datasets. Although summarization systems are usually evaluated on newswire text, research community of this application domain also used other corpora (e.g. the IMDB movie review sentiment corpus), in particular with deep learning techniques (Denil et al. 2014). We also demonstrate the application of the domain other than newswire. Unlike newswire text, emails have more variety of structures, tend to be less formal and do not undergo any meticulous editorial process. Also, each email may have a separate topic whereas in most newswire-based corpora the number of topics is more restricted. With evaluating the system using two different datasets, developed from three different email sources, the system could be extensively explored.

Using the word "automatic" in text summarization goes back to (Luhn 1958). Luhn (1958) appears to have introduced the first known machine-based summarization mentioned:

> **Abstract**: Excerpts of technical papers and magazine articles that serve the purposes of conventional abstracts have been created entirely by automatic means… **Conclusion**: auto-abstracts have a … as they are the product of a statistical analysis of the author's own words … The auto-abstract is perhaps the first example of a machine-generated equivalent of a completely intellectual task in the field of literature evaluation (Luhn 1958).

A key concept of deep learning, in particular AEs, is "automatically learning features". In fact, in this thesis, the word "automatic" presents different concept with historically using of this word in text summarization community. In the literature "automatic" indicates to a machine-based solution whereas in this thesis the word implies the feature learning process.

1.2 Outline of the Thesis

This thesis is divided into five chapters. The next section of this chapter is a description of contributions of the thesis. Chapter two is literature review including previous investigations, corpora and evaluation metrics. Chapter three is dedicated to our model. First a Restricted Boltzmann Machine (RBM) and AEs will be described in terms of topology and training algorithms. Then our system framework and the word representation will be presented. Chapter four presents results and discussion on the two datasets. In chapter five, the conclusion of the thesis and future work will be presented. Bibliography and appendix are placed at the end of this thesis.

1.3 Contributions of This Research

A brief explanation of the contributions of this thesis is following:

- One key factor of most machine learning techniques for text summarization is relying on labeled training set; however, almost all researchers of this application domain agreed that there is no single best summary for a text, consequently no appropriate labeling. Also, unintentional human mistakes during annotation process is an inevitable issue. Therefore, we introduce an unsupervised approach for extractive summarization using AEs. Although AEs has previously applied for summarization as a word filter (Zhong et al. 2015), to the best of our knowledge we are the first to use the AE for summarization in this sentence ranking fashion.
- There have been many studies to develop word representations, in particular to address with sparsity of the word representation. This thesis explores how an AE can handle sparsity due to word representation and also addresses sparsity with adding random noise in addition to the local word representation. When it comes to adding random noise, the structure changes the application of the AE from a deterministic feed-forward network to a stochastic model.

To our best knowledge, this representation technique is not previously explored in the application of auto-encoders.

- We introduce the Ensemble Noisy Auto-encoder (ENAE) in which the model is trained once and used multiple times on the same input, each time with different added noise. In the thesis, we explore how adding stochastic noise to the input and running an ensemble on the same input can make improvements.
- In email summarization domain, most summarizers rely on features that are completely dependent to human-crafted techniques. They are mostly limited to numerical optimization of weights for human designed features (e.g. position of sentence, subjectivity labels) from the text; one of the key characteristics of our framework, however, is to automatically develop features or representations from the original text. This automatic feature extraction is indeed one of the goals of designing shallow and deep neural network structures (Bengio 2009, Deng et al. 2014). The thesis expands the email summarization features beyond a set of features and develop to automatically extracted features.

Chapter 2

2. Related Work

For the purpose of this chapter, previous studies will be classified based on applied techniques into three categories: classic, machine learning and neural network-based techniques. In this section, the three categories will be reviewed, separately. In fact, the section is based on analyzing different methods of text summarization while presenting a variety of applications. Also, there will be a subsection for reviewing some commonly used corpora and evaluation metrics.

2.1 Classic Methods

Most early summarizers were based on the frequency of the word occurrence in a text as a factor to measure the significance of a sentence to give a score in the ranking phase. The idea was that repeatedly using a word shows the word's significance for the writer.

The language model in the classic summarization techniques was mostly a unigram probability distribution over a word. This model is only based on the frequency of words, so-called bag-of-words (BOW) (see section 3.3), and it ignores the position of the words. Other word and sentence level features were relative position of the sentence and key phrases (e.g. title and heading words). These features have been used for both generic and query-based summarization.

To obtain a generic summary, a commonly used method is to compute a relevance score between each sentence and the whole document (Gong and Liu 2001). For each term, there would be two different term-frequency (*tf*) weightings (see section 3.3), one based on local weighting and the other based on the global weighting. Having the weighted *tf* vector of a sentence and the weighted *tf* vector of the whole document, the inner product of the two vectors will be the relevance score. Sentences with the highest relevance score will be ranked and added into the summary.

The term "relevance" in query-based summarization reflects the relevance of a document or multi-document to a user-specific query. More precisely, a user needs information expressed as a query (e.g. a clinical query: What is the most effective malaria prophylaxis during pregnancy?) and a summarizer provides the most relevant and salient characteristics of the documents according to that query (Berger and Mittal 2000). In addition to *tf* representation, term frequency - inverse document frequency (*tf-idf*) (see section 3.3) has been commonly used in query-relevant summarization. Variations of the *tf-idf* weighting technique are often used for weighting a document and the query (Wu et al. 2008).

In a document, there are probably a set of sentences that convey similar information and each sentence may have high relevance scores; thus, the summary will not cover the major topics of the document, but only a content in the different sentences. Minimizing redundancy in the summary is an important phase for both query-based and generic summarization. In particular, multi-document summarization requires a redundancy phase. To tackle this deficiency, one simple but effective technique is that when a sentence get selected to be added into the summary, not only the sentence has to be eliminated from the document, but all the terms of the sentence also have to be deleted from the document. This procedure ensures that the subsequent sentence selection will convey new information and have minimum overlap with the previous selected sentences. Another robust technique is that once a system extracts a set of sentences, in which sentences carry somewhat similar information (e.g. using clustering), one of these sentences can be chosen to represent the set. Barzilay and McKeown (2005) presented more complicated approaches. They are more applicable to an abstractive summarization.

Maximum marginal relevance (MMR) is a commonly used technique to ensure both minimum redundancy and maintaining relevance for ranking (Carbonell et al. 1998). In MMR a linear combination of relevance and information-novelty has been proposed and this combination is called "marginal relevance". The approach tries to maximize the combination to simultaneously have relevant sentence to the query and contain minimal similarity to previous selected sentences. This algorithm is applied in the sentence selection phase. After ranking sentences in descending order in accordance with their relevance scores (with a given threshold), sentences are selected iteratively to put into the summary. Each time the sentence will be added to the summary if it is not significantly similar to sentences already picked for the summary, otherwise the sentence has to be put aside. The iteration is continued up until the limitation of the summary being met. More recently, McDonald (2007) proposed a dynamic programming approach based on solutions to the knapsack problem; he replaced, the greedy search of MMR with the dynamic programming approach in which optimal accuracy and scaling properties have been guaranteed.

Clustering and graph-based models have also been successfully adapted and used for text summarization. For example, centroid models such as the k-means algorithm and MEAD (a centroid-based summarization using a feature-based generic summarization toolkit (Radev et al. 2000)) have presented promising results. In particular, MEAD uses the centroids of the clusters to identify which sentences are central to the topic of the cluster. The centroids of the clusters could be produced using modified *tf-idf* feature vectors. Erkan and Radev (2004) introduced several other criteria to assess sentence centrality (or salience) on the basis of a graph modeling. These criteria were established to find the sentences (as salient) that are similar to many of the other sentences in a cluster rather than the centrality of the words that sentence contains.

2.2 Machine Learning Approaches

Machine learning summarizers are trainable systems in which models learns how to generalize its parameters to extract salient points. The feature sets in the techniques are mostly the same in classic methods alongside other features (e.g. features based on bigram (N-gram) language model). Most of machine learning approaches in text summarization are inspired from information retrieval and adapted into the task such as Bayesian models, Hidden Markov Models (HMM), Support Vector Machines (SVM) and Support Vector Regression (SVR) (Cortes and Vapnik 1995, Manning and Schütze 1999). In general, many supervised and unsupervised approaches have been proposed and they can be categorized into the following groups: Latent topic models as unsupervised techniques, classification and regression as the supervised techniques.

Latent topic models such as probabilistic Latent Semantic Analysis (pLSA) and Latent Dirichlet Allocation (LDA) have been widely used in text summarization (Hennig 2009, Li and Li 2014). Most LDA and pLSA based summarizers are being trained in an unsupervised manner. In pLSA, the probability of each co-occurrence of words and documents is being considered a mixture of conditionally independent distributions. On the other hand, Blei et al. (2003) introduced LDA. It is a three-level hierarchical Bayesian model and the topic is assumed to have a Dirichlet prior. One key point is that the pLSA model can be considered equivalent to the LDA model under a uniform Dirichlet prior probability distribution.

HMM has been used as a statistical model for text summarization. In contrast to a Naive Bayes Classifier that has independency assumption, the HMM has fewer assumptions. In the naive Bayes, the probability of a sentence to be chosen to add into summary is independent of whether previous sentence is in the summary. Additionally, a Naive Bayes Classifier has the assumption that features are all independent. However, in HMM (Conroy and O'leary 2001) independency of selected sentences was not assumed and a joint distribution of the feature set was used.

SVM has been another supervised approach for setting up a summarizer. Li et al. (2009) proposed the graph structure of the output variables and also employed structural SVM to solve its optimization problem. The notion was to provide a structural SVM with three constraints to have more diversity, coverage and balance in the summary.

A motivation to use Applying Support Vector Regression (SVR) for text summarization is that it can estimate the importance of sentences with providing continuous values. More clearly, the regression type of SVM can rank sentences with continuous values instead of putting sentences into classes. Ouyang et al. (2011) developed a regression model for query-focused multi-document summarization. It uses SVR to learn a sentence scoring function with the proposed feature set of the study.

2.3 Artificial Neural Network Models

Neural networks are trainable statistical models based on an analogy with the structure of the human brain. Different classifications of artificial neural networks haven been presented on the basis of training algorithms, topologies and applications (Anderson 1995, Bishop 1995, Hagan et al. 1996, Bengio 2009, Deng et al. 2014, Bengio 2015). One commonly used classification is based on how deep a neural network is. To be more exact, a neural network with only one hidden layer is considered as a shallow neural network and with more than one hidden layer as a deep structure. The main focus of this section is deep architectures.

2.3.1 Text Summarization Using Shallow Neural Networks

Kaikhah (2004) successfully introduced a shallow neural network for automatic text summarization. The summarizer had three phases: general training to learn the relevant characteristics of sentences, prune the neural network to eliminate the unimportant features and select highly ranked sentences. This system motivated other researchers to use artificial neural networks for summarization. Kulkarni and Prasad (2010) proposed a combination of the multilayer perceptron (MLP) with fuzzy logic and they reported that this combination improves the result. Also, Krysta Svore (2007) proposed a neural-network-based system, called NetSum. This technique was inspired from (Burges et al. 2005). All mentioned systems use the same feature level (e.g. sentence position, sentence length, title similarity score), have supervised training algorithm and feed-forward structure.

In addition to feed-forward neural networks, recurrent neural networks (RNNs) have been applied for text summarization (Prasad 2009). In spite of different topology, the approach was generally inspired from (Kaikhah 2004).

2.3.2 Text Summarization Using Deep Learning

The concept of deep learning can be considered as a wide class of machine learning approaches and architectures in which the main characteristic is hierarchically using many layers (i.e. more than one hidden layer) of nonlinear information processing. The aim of the techniques is learning feature hierarchies with higher level features of the hierarchy extracted from the composition of lower level features (Bengio 2009). In fact, with automatically learning features at multiple levels of abstraction a system may execute complex functions to directly transfer the input to the output, completely independent from human-crafted features.

The successful application of deep neural networks to speech, image, vision and NLP tasks shows the capability and flexibility of the idea. In particular, Arisoy et al. (2012) proposed a deep structure for language modeling (LM) where the aim is to assign a probability to any arbitrary sequence of the words (or other linguistic symbols such as letters, characters, phonemes, etc.). Also, application to NLP is currently one of the most promising areas in deep learning research. Although NLP also deals with sequences of words, the task is highly diverse and not just focusing on assigning probabilities for linguistic symbols. One example of successfully applying deep learning for NLP tasks was presented by Collobert et al. (2011). They introduced a semi-supervised learning method for the shared tasks of part-of-speech tagging, chunking, named entity recognition, and semantic role labeling. All the tasks were integrated into a single system to be trained jointly.

Applying deep neural networks to text summarization has shown promising results (Liu et al. 2012, PadmaPriya and Duraiswamy 2013). The intersection between the studies is the application of the Restricted Boltzmann Machine (RBM) (see section 3.1.1). A RBM is a generative model (i.e. learning a join probability distribution over observation and hidden variables) with one layer of (typically Bernoulli or Gaussian) stochastic visible (or observable) units and one layer of (typically Bernoulli) stochastic hidden units. However, the difference between (Liu et al. 2012) and (PadmaPriya et al. 2013) is to some extend fundamental. PadmaPriya et al. (2013) used the sentence level features while Liu et al. (2012) applied *tf* representation.

In the seminal paper of (Bengio et al. 2003), a new neural-network-based LM technique was proposed. The proposed method provides a distributed representation for words called "word embeddings". The representation is now a crucial part of many applications such as extractive summarization (Kågebäck et al. 2014). Indeed, in contrast to n-gram language models in which words are treated as discrete entities, the neural network language model (NNLM) embeds words in a continuous space. This estimation was performed by a feed-forward or recurrent shallow neural network. More recently, deep neural network language models (DNN LMs) offers improvements over the NNLM (Arisoy et al. 2012). It is claimed that a distributed representation of a word is in fact a vector of features which characterizes the meaning of the word; however, it is still an open question.

After successful application of word embeddings, researchers have been developing vector representations for different pieces of text such as phrases, sentences and paragraphs (Palangi et al. , Tomáš 2012, Mikolov et al. 2013, Le and Mikolov 2014). One of the main motivation is that word embedding is indifferent to the word order and also it is not easy to provide a fixed-length representation for a document. Sentence embeddings can mostly overcome the deficiency of word embedding; however, it is not clear how well the vector space reflects the documents semantic meaning or discovers salient words and topics in a sentence. Paragraph embeddings give rise to even more complex questions.

As mentioned earlier, deep learning has been used in a wide variety of applications and consequently they have mutually inspired one another. One example of this inspiration is convolutional neural networks (CNNs). Denil et al. (2014) proposed a novel visualization approach, inspired from CNNs for computer vision, that shows promising results to produce a compelling automatic summarization system for documents. It is true that CNNs are in nature able to handle variable-length input, but the input of the CNN is word embeddings and the system tries to work well as a review sentiment classifier. The task of the model can be divided into two phases: word embeddings to sentence embeddings and sentence embeddings to document embeddings. Similar to CNNs application in computer vision, the system is a supervised technique; it is trained to classify sentiment labels that are either positive or negative (i.e. a binary label).

Recursive neural networks (RNNs) in which the same set of weights applies recursively have been successfully used in NLP. Quite recently this type of deep neural networks has been used for text summarization (Cao et al. 2015). In fact, with the capability of dealing with a variable-length input in a RNN, the proposed system formulates the sentence ranking task in a hierarchical regression fashion. It measures the salience of a sentence and its constituents (e.g., phrases), simultaneously. The model is a supervised technique and uses hand-crafted word features as inputs.

2.4 Summarization Corpora

It is quite important to have a trustable dataset in text summarization since the most challenging part of providing a dataset for this task is manual annotations. Indeed, not only the developers have to handle privacies, domain specific issues and the cost of the project, also annotators should be well trained both for extractive and abstractive tasks. In spite of all these difficulties, the summarization community produced a wide variety of datasets for general and domain specific applications. For example, Document Understanding Conference[1] (DUC), Text Analysis Conference[2] (TAC) and Text REtrieval Conference[3] (TREC) are among the most popular corpora for text summarization. They all comprise newswire articles.

Many research groups developed domain specific corpora, for instance, evidence-based medicine summarization (Molla and Santiago-Martinez 2011), to achieve their specific goals. Databases

[1] http://duc.nist.gov/

[2] http://www.nist.gov/tac/

[3] http://trec.nist.gov/data.html

consist of meeting records were also developed for summarization (e.g. Augmented Multiparty Interaction[1] (AMI)).

The most popular email dataset is the Enron corpus[2]. This uncensored corpus, that was made public during the legal investigation into the Enron Corporation, is a treasure for a wide variety of purposes. Since Enron contains over 300,000 threads only a part of the corpus was manually annotated for summarization in the University of British Columbia and it is not publicly available. However, other research groups have developed and manually annotated email datasets that are publicly available such as the British Columbia Conversation Corpus[3] (BC3) and Summarization and Keyword Extraction from Emails[4] (SKE).

Although datasets on which almost all summarization research studies have evaluated are developed specifically for summarization, Denil et al. (2014) used a sentiment analysis dataset for text summarization. Indeed, Denil et al. (2014) implicitly argued that during the prediction of a sentiment of a movie reviewer, text summarization can be defined.

2.5 Evaluation Metrics

Unlike many NLP tasks, there is no single golden-standard evaluation metric for summarization. In addition to manual techniques, there are standard machine-based evaluation approaches. The goal of all metrics is to determine how much the generated summary is similar to human generated one for both extractive and abstractive summarization.

The most trusted evaluation metric is Recall-Oriented Understudy for Gisting Evaluation (ROUGE) as a set of automatic metrics and a software package (Lin 2004). This metric counts the overlap units such as unigram and bigram and therefore it is based on n-gram similarity between the model summary (i.e. human generated) and the system summary (i.e. machine generated). The ROUGE package provides the average Precision, Recall and F-score:

$$\text{precision} = \frac{|\{\text{relevent sentences}\} \cap \{\text{retrieved sentences}\}|}{|\{\text{retrieved sentences}\}|} \qquad (2.1)$$

$$\text{recall} = \frac{|\{\text{relevent sentences}\} \cap \{\text{retrieved sentences}\}|}{|\{\text{relevent sentences}\}|} \qquad (2.2)$$

$$\text{F-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \qquad (2.3)$$

[1] http://www.amiproject.org/
[2] https://www.cs.cmu.edu/~./enron/
[3] https://www.cs.ubc.ca/cs-research/lci/research-groups/natural-language-processing/bc3.html
[4] http://web.eecs.umich.edu/~mihalcea/downloads.html

Precision index shows how many extracted items are relevant and recall index shows how many relevant items are extracted by the model. F-score is a weighted harmonic average of precision and recall and an index of a test's accuracy.

ROUGE has different measures: ROUGE-N (i.e. N-gram Co-Occurrence Statistics), ROUGE-L (i.e. Longest Common Subsequence), ROUGE-W (i.e. Weighted Longest Common Subsequence), and ROUGE-S (i.e. Skip-Bigram Co-Occurrence Statistics). Although accepting this metric across all domains has not been established, (Dang and Owczarzak 2008) observed that among all ROUGE measures ROUGE-2 recall has the highest correlation with manual scores.

Deep learning shows promising results in NLP and developing an evaluation metric for summarization using deep neural network is one of these outcomes. Genest et al. (2011) introduced an automatic approach to score a summary. This structure using both unsupervised and supervised training algorithm gave a good result for each individual summary, but not in the average scores. Also, in an unconventional approach, Denil et al. (2014) introduced a new quantitative measure for summary evaluation. They argue that without any labeled (i.e. human annotated sentences), correct sentence selection can result in the easier identification of a movie reviewer's sentiment (i.e. a label); therefore, the sentence has to be selected for the summary.

With the assumption of having no single best summary, Pyramid method (Nenkova et al. 2007) is a predominant quantitative evaluation. It is an indirect manual evaluation in which human content selection variations incorporate in the evaluation. Indeed, this technique relies on agreement of the human summaries. More precise, summary content units (SCUs) extracted from the models (i.e. multiple human generated summaries) and appeared in most of the models are weighted more highly. The SCUs are mostly no bigger than a clause. Then, the SCUs are arranged in a pyramid architecture and provide pyramid score.

Chapter 3

3. The Methods and Algorithms of the Architecture

The thesis explores two different summarization settings in which a document is summarized based on a written text not included in the original text. Thus, the algorithm can be considered as an extractive query-oriented summarization.

This chapter describes the architecture of our model. The AE is explained, in detail, in the first part. The section elaborates on topology and mathematic of pre-training and fine-tune phases. In the second section, regularization method is described. Section three presents the word representations as the input of the AE. The sentence ranking phase is described in the fourth section. In the last section, iterative data fusion and voting that is indeed sentence selection phase is presented.

3.1 The Deep Auto-encoder

AE neural networks are a particular type of unsupervised feature (or representation) learning technique in which a transformation of original data to a representation that is proper for a machine learning task is required. In representation learning, the system can learn for a specific task and the features themselves, both automatically.

An AE is a feed-forward network that learns to reconstruct the input *x* (Hinton and Zemel 1997). More precise, it is trained to encode the input *x* using a set of recognition weights into a feature space *C(x)*. Then, the features (codes) *C(x)* are converted into an approximate reconstruction of $\hat{x}$ using a set of generative weights. The generative weights are indeed obtained firstly from unrolled weights of encoder and then from fine-tune phase. As illustrated in figure 3-1, through the encoder (i.e. mapping to a hidden representation) the dimensionality of *x* is reduced to the give number of codes. The codes are then mapped to $\hat{x}$ through the decoder. Both non-linear mappings are deterministic. Because there is no labeled training data, the algorithm is an unsupervised learning.

Each layer has its own topological structure and although coding layer (a.k.a discriminative layer or bottleneck) usually consists of a reduced number of units, other hidden layers may have more units than the input. What a deep AE performs is transforming the original input to a better representation over a hierarchical features or representations, each level of this hierarchy corresponding a different level of abstraction.

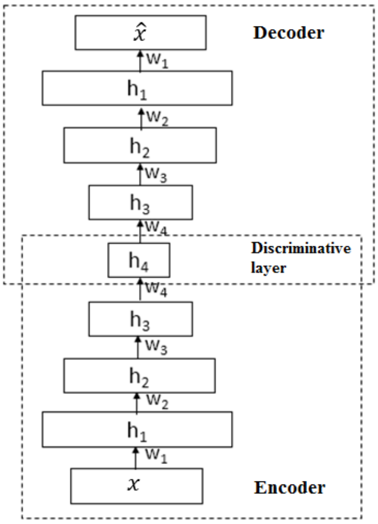


Figure 3-1 the structure of an AE for dimensionality reduction. The weights of decoder are obtained from unrolled weights of encoder. x and $\hat{x}$ denote the input and reconstructed inputs respectively. h_i are the hidden layers and w_i are the weights. Features (codes) C(x) are in this instance extracted from the output of the hidden layer h_4.

Although stacking linear projections can end up a lower-dimensional representation, this model is not a deep structure and can be seen as a principal component analysis (PCA) (Baldi and Hornik 1989). In general, when an AE has linear activation function and the number of hidden units are less than the input dimension, the AE projects a subspace of the principal components of the input. However, it is expected the AE with non-linear activation functions learns more useful feature-detectors (Hinton and Salakhutdinov 2006a). Also, in most deep AE and in our case the number of first layer units is more than the input dimension, expecting a richer representation in the layer.

It is true that early algorithms were successful in accomplishing a given task using AEs, the network took a long time to be trained. Another weak point of early approaches was that the back-propagated gradients of the error were very small when they arrive at the lower layers (Hinton et al. 2006a). This phenomenon is known as the vanishing back-propagated gradients. Following the introduction of deep architectures, many researchers tried to solve the mentioned problems but failed until the mid-2000s. Hinton et al. (2006b) introduced an algorithm that was fast enough and could address the problem of vanishing back-propagated gradients. The approach uses an unsupervised method in which a new algorithm was introduced for pre-training followed by a fine-tuning training phase. AEs with pre-training have been successfully applied primarily to documents retrieval (Salakhutdinov and Hinton 2009, Genest et al. 2011). This thesis explores applying this technique to shorter texts.

In contrast to the commonly used fine-tune phase, Huang et al. (2004) presented an algorithm called "extreme learning machine" (ELM) and showed that all the parameters of a single-hidden layer feed-forward network do not need to be tuned and hidden nodes/neurons can be randomly chosen. This approach has been called the extreme learning machine (ELM) since training a single-hidden layer feed-forward network with this algorithm can be very fast. Although one may criticize that ELM is not good for highly complex tasks compared with back-propagation based deep structures, ELM has a promising future (McDonnell and Vladusich 2015).

Our model has two training stages: pre-training and fine-tune. Pre-training is an approach to find an appropriate starting point for the fine-tune phase. That is, obtained parameters in pre-training phase will be the initial weights of the fine-tune phase. This initialization could significantly improve the performance of an AE in a wide variety of applications compared with random initialization (Hinton et al. 2006a, Hinton et al. 2006b).

3.1.1 Pre-training Phase

The observations (i.e. the input of the network) as produced samples from a stochastic generative model can be modeled using a generative model such as Restricted Boltzmann Machine (RBM) (Hinton 2002, Hinton 2010). The RBM, demonstrated in figure 3-2, is the undirected graphical model that can be considered as a two-layer neural network with one layer of observable units and one layer of hidden units (i.e. feature detectors). The weighted connections are restricted between hidden units and visible units, symmetrically, and there are no connections between units of the same layer. Depending on the function of the network, in addition to the exponential family units presented by Welling et al. (2004), the visible and hidden units could be considered as Bernoulli, Gaussian or Multinomial, binomial, rectified linear. The main focus of the thesis is on Bernoulli-Bernoulli units and Gaussian-Bernoulli units.

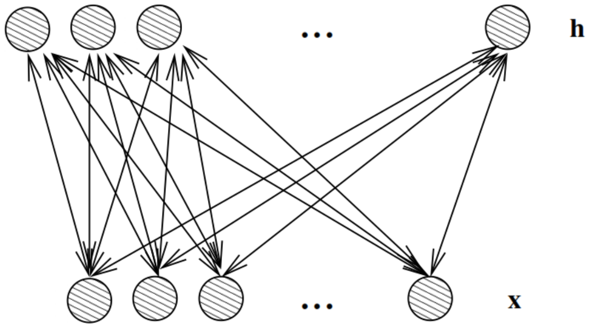


Figure 3-2 the structure of Restricted Boltzmann Machine (RBM) as an undirected graphical model.

For binary states of visible units and binary feature detectors (i.e. Bernoulli-Bernoulli units) and in this energy-based network, the energy function (Hopfield 1982) is bilinear:

$$E\ (\boldsymbol{x}, \boldsymbol{h}; \theta) = -\sum_{i \in \text{visible}} b_i x_i - \sum_{j \in \text{hidden}} a_j h_j - \sum_{i,j} x_i h_j w_{ij} \tag{3.1}$$

Where $w_{i,j}$ is the weights between visible units x_i and hidden units h_j, b_i and a_j are their biases. In terms of the energy function the joint distribution $p(\boldsymbol{x}, \boldsymbol{h}; \theta)$ has following equation:

$$p(\boldsymbol{x}, \boldsymbol{h}; \theta) = \frac{\exp(-E(\boldsymbol{x}, \boldsymbol{h}; \theta))}{Z} \tag{3.2}$$

Where $Z = \sum_{\boldsymbol{x},\boldsymbol{h}} \exp(-E(\boldsymbol{x},\boldsymbol{h};\theta))$ is a normalization factor (a.k.a the partition function). The marginal probability of assigning to a visible vector is:

$$p(\boldsymbol{x};\theta) = \frac{\sum_{\boldsymbol{h}} \exp(-E(\boldsymbol{x},\boldsymbol{h};\theta))}{Z} \tag{3.3}$$

Based on above mentioned equations and because of symmetric structure of the network the conditional probabilities for Bernoulli-Bernoulli RBM:

$$p\left(h_j = 1|\boldsymbol{x};\theta\right) = \frac{\exp(\sum_i w_{ij}x_i + a_j)}{1+\exp(\sum_i w_{ij}x_i + a_j)} = f(\sum_i w_{ij}x_i + a_j) \tag{3.4}$$

$$p(x_i = 1|\boldsymbol{h};\theta) = \frac{\exp(\sum_j w_{ij}h_j + b_i)}{1+\exp(\sum_j w_{ij}h_j + b_i)} = f(\sum_j w_{ij}h_j + b_i) \tag{3.5}$$

When visible units are real values and hidden units binary, the energy function becomes:

$$E\ (\boldsymbol{x},\boldsymbol{h};\theta) = \sum_{i\in\text{visible}} \frac{(x_i - b_i)^2}{2{\sigma_i}^2} - \sum_{j\in\text{hidden}} a_j h_j - \sum_{i,j} \frac{x_i}{\sigma_i} h_j w_{ij} \tag{3.6}$$

Where σ_i is the standard deviation. With unit-variance for this Gaussian-Bernoulli conditional probabilities are:

$$p\left(h_j = 1|\boldsymbol{x};\theta\right) = \frac{\exp(\sum_i w_{ij}x_i + a_j)}{1+\exp(\sum_i w_{ij}x_i + a_j)} = f(\sum_i w_{ij}x_i + a_j) \tag{3.7}$$

$$p(x_i|\boldsymbol{h};\theta) = \frac{1}{\sqrt{2\pi}} e^{\left(-\frac{\left(x - b_i - \sum_j w_{ij}h_j\right)^2}{2}\right)} = N(\sum_j w_{ij}h_j + b_i\,, 1) \tag{3.8}$$

To estimate the parameters of the network, maximum likelihood estimation (equivalent to minimizing the negative log-likelihood) can be applied. Taking the derivative of the negative log-probability of the inputs with respect to the weights is:

$$\frac{\partial -\log p(x)}{\partial \theta_{i,j}} = \frac{\partial}{\partial \theta}(-\log \sum_{\boldsymbol{h}} P(\boldsymbol{x},\boldsymbol{h})) \tag{3.9}$$

$$= \frac{\partial}{\partial \theta}\left(-\log \sum_{\boldsymbol{h}} \frac{e^{(-E(x,\boldsymbol{h}))}}{Z}\right) \tag{3.10}$$

$$= -\frac{Z}{\sum_{\boldsymbol{h}} e^{(-E(x,\boldsymbol{h}))}}\left(\sum_{\boldsymbol{h}} \frac{1}{Z}\frac{\partial e^{(-E(x,\boldsymbol{h}))}}{\partial \theta} - \sum_{\boldsymbol{h}} \frac{e^{(-E(x,\boldsymbol{h}))}}{Z^2}\frac{\partial Z}{\partial \theta}\right) \tag{3.11}$$

$$= \sum_{\boldsymbol{h}}\left(\frac{e^{(-E(x,\boldsymbol{h}))}}{\sum_{\hat{\boldsymbol{h}}} e^{\left(-E(x,\hat{\boldsymbol{h}})\right)}}\frac{\partial E(x,\boldsymbol{h})}{\partial \theta}\right) + \frac{1}{Z}\frac{\partial Z}{\partial \theta} \tag{3.12}$$

$$= \sum_{\boldsymbol{h}} P(\boldsymbol{h}|\boldsymbol{x})\frac{\partial E(\boldsymbol{x},\boldsymbol{h})}{\partial \theta} - \frac{1}{Z}\sum_{\boldsymbol{x},\boldsymbol{h}} e^{(-E(\boldsymbol{x},\boldsymbol{h}))}\frac{\partial E(\boldsymbol{x},\boldsymbol{h})}{\partial \theta} \tag{3.13}$$

$$= \sum_{\boldsymbol{h}} P(\boldsymbol{h}|\boldsymbol{x})\frac{\partial E(\boldsymbol{x},\boldsymbol{h})}{\partial \theta} - \sum_{\boldsymbol{x},\boldsymbol{h}} P(\boldsymbol{x},\boldsymbol{h})\frac{\partial E(\boldsymbol{x},\boldsymbol{h})}{\partial \theta} \tag{3.14}$$

$$\frac{\partial -\log p(x)}{\partial \theta_{i,j}} = E\left[\frac{\partial E(x,\boldsymbol{h})}{\partial \theta}|\boldsymbol{x}\right] - E\left[\frac{\partial E(x,\boldsymbol{h})}{\partial \theta}\right] \tag{3.15}$$

$$\frac{\partial -\log p(\boldsymbol{x})}{\partial \theta_{i,j}} = < x_i h_j >_{data} - < x_i h_j >_{recon} \tag{3.16}$$

The angle brackets denote the expectation of the distribution of the subscriptions. This leads to a learning algorithm in which the update rule for weights of a RBM is given by:

$$\Delta w_{ij} = \varepsilon(< x_i h_j >_{data} - < x_i h_j >_{recon}) \tag{3.17}$$

Where ε is a learning rate, $< x_i h_j >_{data}$ is the so-called positive phase contribution and $< x_i h_j >_{recon}$ is the so-called negative phase contribution. In particular, the positive phase is trying to decrease the energy of observation and negative phase increase the energy defined by the model.

Although it is not easy to compute the expectation defined by the model, the k-step contrastive divergence (CD-k) approximation provides surprising results. In practice, Hinton (2002) presented that maximizing the likelihood or log-likelihood of the data is equivalent to minimize Kullback–Leibler (KL) divergence between data distribution and the equilibrium distribution over the visible variables. Thus, to avoid the computational complexity of the log-likelihood gradient, the contrastive divergence (CD) method, which approximately follows the gradient of the difference of the two KL divergences, can be used. The approximation is based on Gibbs sampling and this sampling procedure can be shown as such figure 3-3.

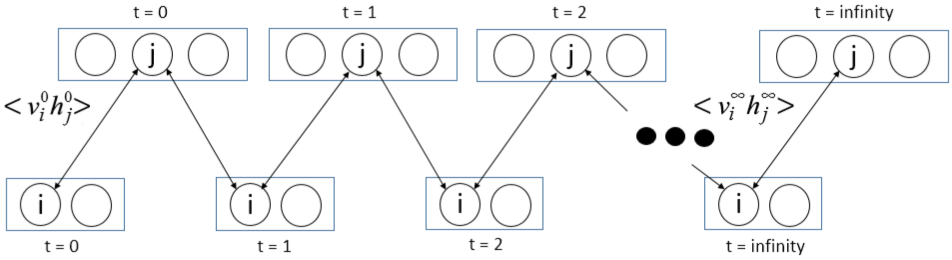


Figure 3-3 Gibbs sampler for an infinity sampling steps (figure redrawn after figure 4 of Hinton et al. (2006b)).

There has been numerical comparison between calculating the update rule with exact log-likelihood gradient and CD-k (Carreira-Perpinan and Hinton 2005). To have a very good approximation and low computational complexity we used CD-1, with running one step Gibbs sampler that is:

$$\boldsymbol{x} = \boldsymbol{x}^0 \xrightarrow{P(\boldsymbol{h}|\boldsymbol{x}^0)} \boldsymbol{h}^0 \xrightarrow{P(\boldsymbol{x}|\boldsymbol{h}^0)} \boldsymbol{x}^1 \xrightarrow{P(\boldsymbol{h}|\boldsymbol{x}^1)} \boldsymbol{h}^1 \tag{3.18}$$

To draw a conclusion, learning was done with 1-step Contrastive Divergence.

The RBM blocks can be stacked to form the topology of the desired AE. During pre-training, the AE is trained in a greedy layer-wise fashion using individual RBMs, where the output of one trained RBM is used as input for the next layer RBM (Figure 3).

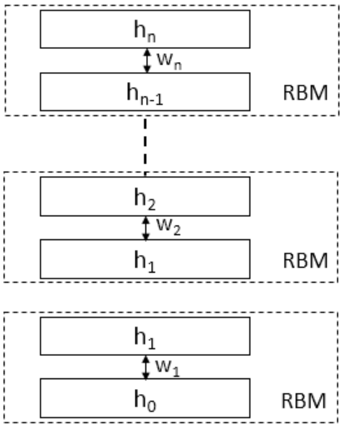


Figure 3-4 Stacking n generative RBM model on top of each other.

In this thesis, first RBM is Gaussian-Bernoulli and the other RBMs are Bernoulli-Bernoulli. More precisely, with a corresponding topology of AE, the weights obtained from stacked RBMs are used as the initial weights of the AE. Erhan et al. (2009) showed that the pre-training process of the first layer can even be as effective as a fully pre-training of the network. Authors argued that empirical simulations showed this procedure. However, our pre-training phase is for the all layers.

3.1.1 Global Adjustment

In this phase, the weights obtained from the greedy layer-wise pre-training are used as an initialization point of AE with a corresponding configuration. In terms of topology, stacking RBM on top of each other and unrolling them (i.e. tied weights) to have a fine-tune phase is the global adjustment of the model. Our AE has the fine-tune phase. The phase is for the entire network (i.e. all the layers together) by a gradient-based optimization algorithm (e.g. Conjugate gradient (CG)). In this unsupervised fine-tuning phase, given the encoding *C(x)*, the whole network tries to minimize the cross-entropy error as the loss function using (BP) algorithm:

$$-\log P(\boldsymbol{x}|c(\boldsymbol{x})) = -\frac{1}{N}\sum_{i=1}^{N} x_i \log \hat{x}_i - \sum_{i=1}^{N}(1-x_i)\log(1-\hat{x}_i) \qquad (3.19)$$

Where n is the total number of items of training data, x is the input of the AE, $\hat{x} = f_\theta(c(x))$ is the reconstructed values.

Over the training phase, the algorithm optimizes weights by minimize the overall loss function, that is:

$$\theta^* = \arg\min_\theta \text{Ł}(f_\theta, \boldsymbol{x}) \qquad (3.20)$$

$$\text{Ł}(f_\theta, \boldsymbol{x}) = \sum_i L(\boldsymbol{x}, f_\theta(c(\boldsymbol{x}))) + \gamma\Omega(\theta) \qquad (3.21)$$

Where f_θ is the parametric function of the network, L is the loss fuction, γ is penalty factor which has been a fixed value in the thesis and $\Omega(\theta) = \sum_{i,j} w_{ij}^2$ is penalty term. The second term is a regularization term (a.k.a weight decay term).

To back-propagate the gradient, the derivative of the cross-entropy function can be written as:

$$\frac{\partial - \log \mathrm{P}(\boldsymbol{x}|c(x))}{\partial \theta} = -\frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i}{\hat{x}_i} - \frac{(1-x_i)}{1-\hat{x}_i}\right)\frac{\partial \hat{x}_i}{\partial \theta} \qquad (3.22)$$

The equation can be simplified to become:

$$\frac{\partial - \log \mathrm{P}(\boldsymbol{x}|c(x))}{\partial \theta} = -\frac{1}{N}\sum_{i=1}^{N}(\hat{x}_i - x_i) \qquad (3.23)$$

The equation shows that the rate of parameters' optimization has a direct relation to the error between desire output and the output.

Briefly, the equations of back-propagation are:

$$\delta^L = \nabla_a C \bullet f' \quad (\mathrm{z}^L) \qquad (3.24)$$

$$\delta^l = ((w^l)^T \delta^{l+1}) \bullet f'(\mathrm{z}^l) \qquad (3.25)$$

$$\frac{\partial C}{\partial b^l} = \delta^l \qquad (3.26)$$

$$\frac{\partial C}{\partial w^l} = a^{l+1}(\delta^l)^T \qquad (3.27)$$

Where C is cost function (here cross-entropy), a is the output of neurons, f is activation function (here sigmoid) and $\nabla_a C = \frac{\partial C}{\partial a^l}$, $z^l = w^l a^{l-1} + b^l$. δ^l is the so-called "error term". w is weight matrix and b is the bias matrix for each layer, l where $l = L-1, L-2, \ldots, 2$. I used "●" to denote the element-wise product operator.

In general, gradient descent algorithms (e.g. back-propagation) take a sequence of steps in weight space. The step comprises a step size and a direction. A simple approach uses a small constant for the step size. However, assuming a properly selected direction, line search algorithm is a commonly used technique to obtain the step size. To guarantee the convergence to the minimum in a finite number of steps or make the process faster, the search direction in weight space can also be effective. Conjugate gradients in which the new step direction can be a compromise between the new gradient direction and the previous search direction (i.e. the search direction and the gradient direction are orthogonal) has been developed and has been widely used for optimization. In conjugate gradients algorithm of this thesis, the Polak-Ribiere rule is used to compute search directions (Fletcher and Reeves 1964, Shewchuk 1994).

In practice, the network parameters can be updated using mini-batch. Indeed, when a gradient-based learning algorithm rely on a simple sample, the variance in the parameter update can make the convergence unsmooth; thus, with picking a mini-batch of the dataset, the algorithm tend to average more training samples thereby smoother convergence.

Although stochastic gradient descent method (SGD) (a.k.a online gradient descent) is a common methodology to fine-tune a neural network, it has a point. The main key disadvantage is difficult tuning. That is, parameters like learning rates have to be tuned by multiple running the algorithm and picking the one that provides the highest performance. Ngiam et al. (2011) evaluated different types of optimization algorithm included SGD and CG. It has been observed that mini-batch CG with line search can simplify and speed up different types of AEs compared to SGD.

3.2 Regularization Method

Applying a progressively complex model results in a wide range of freedom and consequently the model is adapted to a small change of input data. That is, when a model is more complex than necessary it can be more sensitive to small changes. This small change is indeed the difference between train set and test set. One way to address this situation is regularizing the model, that is, putting constraint on the model parameters to have a less degree of freedom. Figure 3-5 demonstrates the relation between error and complexity of a model. It shows minimizing the training error does not necessarily optimize the model for new information.

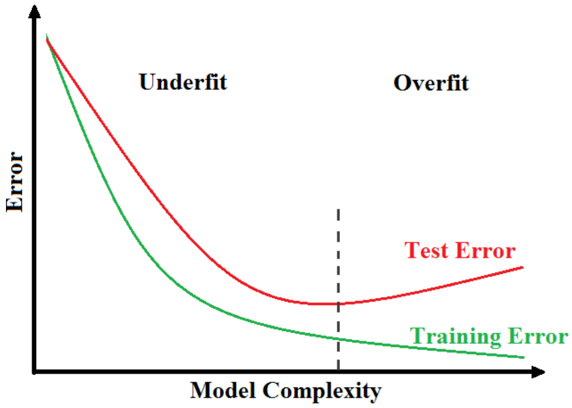


Figure 3-5 the relation between error and complexity of a model.

In the deep structures with millions of parameter, one important factor in training process is generalization. That is, after training, the error on train set becomes very smaller, but for test set (i.e. new data) the error is still large. This problem is called over-fitting (as presented in right side of the figure 3-5) and the techniques for addressing it is regularization. For example, in an AE with more hidden units than inputs (so-called over-complete setting), the network only learns the identify function and although it could precisely reconstruct the input, encoding ends up in a useless features and the network does not learn to generalize for the new data.

There have been extensive investigations to introduce more effective and efficient regularization approaches, in particular for AEs. Training algorithms such as sparsity constraint, contractive mapping, dropout, denoising, L2 regularization (i.g. weight-decay) are among the most commonly used techniques to prevent overfitting and have a regularized AE (Vincent et al. 2008, Ngiam et al.

2011, Rifai et al. 2011, Srivastava et al. 2014). Although all mentioned approaches are different training algorithms or constraints, they are mostly known as variants of AE in the research community.

Many mentioned regularization techniques rely on applying a regularization term for the cost function. The thesis uses a range of techniques such as early stopping and L2 regularization to address overfitting. Also, there is evidence that the unsupervised pre-training support better generalization (Erhan et al. 2010).

One example of applied method in the thesis is early stopping. That is, stopping fine-tuning phase before getting into a local optimum. It is true that applying a gradient based algorithm would increase weights gradually, corresponding to a regularized learning process (Collobert and Bengio 2004), there is no guarantee to have a regularized AE using the early stopping method. Additional techniques such as L2 regularization (i.e. penalizing the sum of squared weights) helps guarantee to have a regularized AE.

3.3 Word Representation

A word representation can be seen as a type of math object regarding each word. The representation of each word could be a value or vector. For example, a binary digit can show the word existence or non-existence, or a continuous value can present a probability. In continuous vector space, each dimension usually corresponds to a lexical semantic or syntactic interpretation. Distributional representation (i.e. words with similar distributions can have same meaning thereby same representations), distributed representation (i.e. presenting a latent feature of the word in each dimension of a real-value vector) and bag-of-words (BOW) representation are among the most commonly used representations. All mentioned representations does not contain task-specific features. The term space of this thesis will be the BOW representation.

In the BOW representation, a sentence or a document is presented as the bag of its words ignoring the exact ordering of the words and only retaining the number of occurrences of each word. A characteristic of the variants of BOW representation is providing a fix size vector (each dimension equal to a word of the vocabulary) for documents with different size. It is intuitive that sentences or documents with similar BOW representations can be similar in content. Although BOW losses word order, neural network with this representation as the input can learn a distributed semantic representation by which semantically related words has less distance from each other in the generated vector space (Turian et al. 2010).

The most common BOW representations used in information retrieval and text summarization systems is term frequency - inverse document frequency (*tf-idf*) (Wu et al. 2008). *tf-idf* represents each word in the document using its term frequency *tf* in the document, as well as over all documents (*idf*):

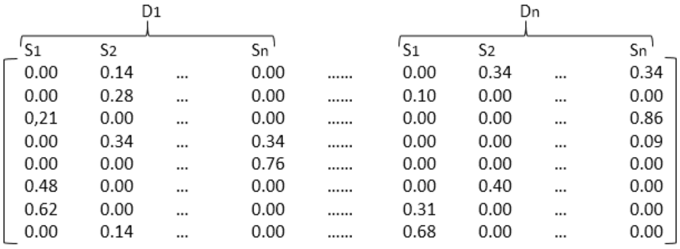
$$\mathrm{tfidf}_{\mathrm{t,d}} = \mathrm{tf}_{\mathrm{t,d}} \times \mathrm{idf}_{\mathrm{t}} \tag{3.28}$$

Where *idf* of a term *t* in document *d* is $\log \frac{N}{df_t}$ (*N* is the total number of documents in a dataset).

In the context of text summarization, the *tf-idf* representations are constructed for each sentence. This means that the input vectors are very sparse because each sentence only contains a small number of words. Sparse representations such as *tf-idf* can cause two problems for the model. First, not observing (enough) data in training process. For example, for *tf-idf* representation with 10-fold cross-validation, on average 6.8% and 5.9% of words of vocabulary is not observed in train set for the SKE and BC3 datasets respectively; therefore, the network has to address words that have never seen over the train process and are just appeared in the test set. Second, too much zero (e.g. for the SKE dataset, 99.5% of the dimensions contain zero with vocabulary size 1000) in input and output of AE. Extensive zeros in the output has effect on the derivative. That is, the small errors on the zeroes can dominate the larger error on the ones when back-propagation reaches the first layer of hidden nodes. Genest et al. (2011) presented a re-sampling algorithm to solve this problem for AE.

However, to address the sparsity, we explore weighted term-frequency (we call it *tf* over the next sections of the thesis) representation using local vocabulary. Due to locally weighting, there will be no unobserved words for network in the train set and the percentage of none zero element of input can increase significantly. For example, for the SKE corpus, with vocabulary size 60, 2.3% of the input dimension contains non-zero for *tf-idf* representation and this percentage is 8.3% for the *tf* representation. Also, over experiments we added small values (noise) to the *tf* representation to have no zero in the input. The idea is that when the noise is small, the information in the noisy inputs is essentially the same as in the input vectors without noise, however, the noisy inputs do not contain zeros anymore.

More precisely, in this *tf* representation, the vocabulary for each document is constructed separately from the most frequent words occurring in each document. The representation is also normalized to Euclidean distance of the vocabulary. We use the same number of words in the vocabulary for each document (presented in figure 3-6). This local representation is less sparse compared to the *tf-idf* because the dimensions in the input now correspond to words that all occur in the current document.



D1					Dn			
S1	S2		Sn		S1	S2		Sn
0.00	0.14	...	0.00	......	0.00	0.34	...	0.34
0.00	0.28	...	0.00	......	0.10	0.00	...	0.00
0,21	0.00	...	0.00	......	0.00	0.00	...	0.86
0.00	0.34	...	0.34	......	0.00	0.00	...	0.09
0.00	0.00	...	0.76	......	0.00	0.00	...	0.00
0.48	0.00	...	0.00	......	0.00	0.40	...	0.00
0.62	0.00	...	0.00	......	0.31	0.00	...	0.00
0.00	0.14	...	0.00	......	0.68	0.00	...	0.00

Figure 3-6 locally weighted *tf* representation. In this example, let's assume the total number of words of the document is 50 and assume 8 words occur more than one time. Also, vocabulary size is 8 and the frequency of each word is 2, 4, 3, 5, 11, 7, 9 and 2. Sentence one from document one has 3 words, Sentence two has 4 words, Sentence n has 2 words.

Because the goal is to summarize each single document and only the important factor is the semantic similarity of the words of a given document, the AE can map the term space into a concept space. Also, because this *tf* representation relies only on each document, different databases can be concatenated in the input of the AE for the training process.

To stop overfitting, denoising auto-encoders use noise in the input of AE. In a denoising auto-encoder, in order to prevent learning identity function, thereby capturing important information about the input in hidden layers, the input is corrupted and the network tries to undo the effect of this corruption. The intuition is that this rectification can occurs if the network can capture the dependences between the inputs. However, in our model, the training algorithm adds very small noise to the input but the output is still the same as input. While denoising auto-encoders only use noisy input in the training phase, we use the input representations with added noise both during training and also later when we use the trained model as a summarizer. In general, in the model, the lowest layer represents the *tf* (added with a noise) of a document and the coding layer provides the semantic space.

3.4 Sentence Ranking

Extractive text summarization is also known as sentence ranking. The system ranks the sentences based on the most relevant and salient characteristics of the documents; thus, this ranking process can be categorized into the query-oriented summarization and over the thesis we call the given information: a query.

In the *tf* representation, the query is addressed as an individual sentence. Figure 3-7 illustrates the transformation of the term space into the concept space using the AE. One key element of this ranking process comes from the BOW representation.

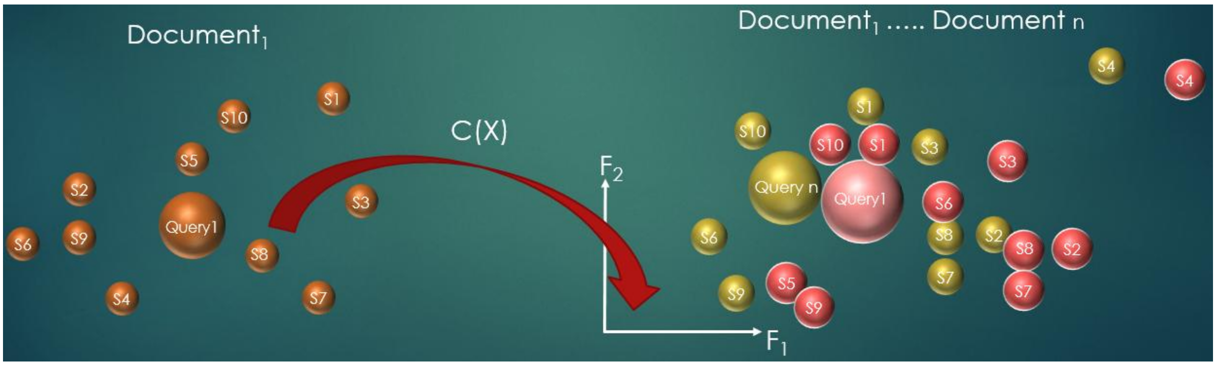


Figure 3-7 the distance between the query and sentences of a document changes on the basis of the semantic mapping.

In the concept space, the cosine similarity, for real values, and Hamming distance, for binary codes, are standard metrics to rank sentences. In a different studies Hinton and Salakhutdinov (2011) and Huang et al. (2013) developed deep structures to provide semantic space for document, but not for sentences, and used Hamming distance and cosine similarity respectively. The AE learns a low-dimensional concept space (i.e. the most essential features of the sentences) $c_Q = C(x|Q)$, $c_s = C(x|S_1, \dots, s_n)$ where *C(x)* is the mapping, Q is the query and S is the sentences. The relevance score, driven from concept codes, between the query (c_Q) and the sentences (c_Q) is calculated by cosine similarity as following:

$$R(Q,D) = cos\left(c_Q, c_s\right) = \frac{c_Q.c_s}{||c_Q||\,||c_s||} \qquad (3.29)$$

Sentences can be ranked according to the relevance score.

3.5 Iterative Data Fusion and Voting

Ensemble methods use multiple models such as classifiers or experts that are combined to solve a particular problem. In general, ensemble models can be categorized into two structures: classifier selection, classifier fusion. In most variants of classifier selection, each individual classifier (also called "expert") is trained in a small regions of feature space whereas in classifier fusion algorithms, all classifiers are trained, in parallel, over the entire feature space (Hansen and Salamon 1990, Breiman 1996, Woods et al. 1996, Fawagreh et al. 2015). The model can be put into the fusion category one.

In our case, after ranking the sentences of a document based on the cosine distance of them, they must be selected to be included into the summary. A straightforward selection strategy adopted in most extractive summarization systems is just to use the top ranked sentences. This selection strategy is for the AE with BOW representation (without additive noise).

However, we explore a more complicated selection strategy that exploits the noisy input representations introduced in the word representation section. By adding random noise to the input, the experiment can be repeated using the same input but with different noise several times, each of which potentially produces a slightly different ranking. Then, an ensemble approach that aggregates the rankings of different experiments on the same input document, each obtained with different added random noise, is used. Finally, the top sentences will be selected according to the final ranking to produce the summary.

In particular, after running the sentence ranking procedure multiple times, each time with different noise in the input, we use a majority voting scheme for fusing the ranks. Since the first rankings have been obtained using a small additive noise (not deterministic input), in the voting algorithm, the first rank of each sentence is not taken into account and only the number of occurrence provides the score.

A detailed schematic of the full model is presented in Figure 3-8. The main difference between our approach and the commonly used ensemble methods lies in the number of applied models. Whereas multiple models are used in the process of ensemble learning, this model only needs to train one model (i.e. an AE) and only one time.

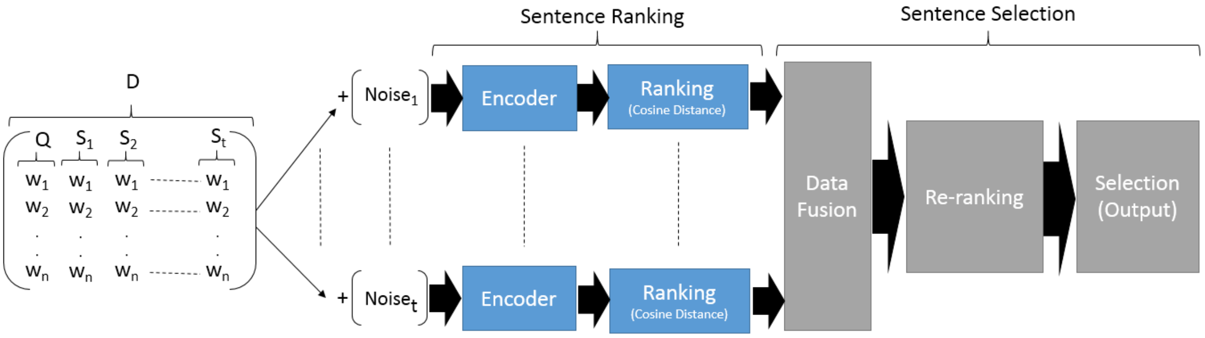


Figure 3-8 the model for noisy input representation.

Thus, in the final sentence selection phase we have a set of items D (the total number of sentences produced by t experiments) and the algorithm picks a subset $S_n \subset D$ (where n is the number of the sentences of the final summary). In a greedy approach $\hat{s}_k$ is picked given:

$$S_{k-1} = \{\hat{s}_1, \dots, \hat{s}_{k-1}\} \qquad (3.30)$$

Where $S_k = S_{k-1} \cup \{\hat{s}_k\}$ with the following formula:

$$\hat{s}_k = \arg\max_{s_k \in D | S_{k-1}} [score(s_k)] \qquad (3.31)$$

Where $score(s_k)$ is the number of times s_k is selected by each component.

The model is an integrated system to automatically rank the sentences semantically related to a query. Indeed, this technique can be considered as an iterative ranking of sentences in which the highest occurrence in the top ranking results in being selected to put into the summary. Although after training, an AE applies a deterministic non-linear transformation to compute a new feature space for the data, the model changes AEs from a deterministic feed-forward network to a stochastic model.

Chapter 4

4. Results and Discussion

This chapter presents the implementation and evaluation of the model. The goal of thesis is the extensive exploration of the model in different conditions. To do so, two different publicly available email datasets, coming from three different email sources, have been selected. In addition to model evaluation, the datasets will be explored using *tf-idf* as a standard and commonly used technique for information retrieval and summarization.

Although summarization systems are usually evaluated on newswire text, research community of this application domain also used other corpora (e.g. the IMDB movie review sentiment corpus), in particular with deep learning techniques (Denil et al. 2014). We also demonstrate the application of the domain other than newswire. Unlike newswire text, emails have more variety of structures, tend to be less formal and do not undergo any meticulous editorial process. Also, each email may have a separate topic whereas in most newswire-based corpora the number of topics is more restricted.

The first dataset, Summarization and Keyword Extraction from Emails (SKE), recently developed for text summarization (Loza et al. 2014). The other dataset was developed in the British Columbia University named BC3 (Ulrich et al. 2008).

Due to conversational structure of email threads writers usually tend to write the gist of required information and shortly. This issue will be explored in this chapter using a sample email. This trend may cause not having redundant sentences for an email thread; thus, this leads us to address the thread emails as single email and ending up in single-document summarization.

We use *tf-idf* as the baseline. One key part of *tf-idf* is the size of vocabulary. There is no general agreement on a vocabulary of a certain size in literature and the application of this representation is a decisive factor. For example, Genest et al. (2011) used less than 1.5% of the unique words of Text Analysis Conference (TAC) corpus, as a very large dataset, for the input of an AE. To provide comparable results, we chose vocabularies with different size but the same size for both datasets. Assuming words with frequency 1 as idiosyncratic words, the longest vocabulary in all experiments can be 1000 words. This size guarantees that the minimum frequency of each word is more than one in both datasets. Also, because *tf-idf* ($v = 1000$) provides the best results compared to *tf-idf* with other vocabulary sizes, we will follow a discussion on *tf-idf* ($v = 1000$) and the models.

We apply the AE model to several different input representations: *tf-idf*, *tf* constructed using local vocabularies, *tf* with added Gaussian or Uniform noise. The number of experiments are 5 for ENAE, that is, 5 experiments of NAE (i.e. training once and 5 times running the experiment with 5 randomly added noise to *tf* representation). We used 10-fold cross-validation for all experiments.

The pre-processing procedure was the same for both corpora. First, in each text, derivatively related words that have similar meaning and come from the same family were altered to a common base form. There were two options for this step: lemmatization (i.e. morphological analysis of words) and stemming (i.e. reducing inflected words to their stem form, but not necessarily identical to the morphological root). Since the thesis only relies on word occurrence, we selected stemming. The Porter stemmer (Porter 1980) was used for this phase. In the second phase, we removed stop-words. Indeed, the most common English words that can influence the word representation as a noise were taken out.

The variance of the additive noise is the same for both Gaussian and Uniform noise over the all experiments. This additive noise is not a deterministic noise that means a noise that is generated once and is used for all data. The additive noise is randomly generated for each individual experiment.

In accordance with the task, the training parameters of the AE have been assigned and were constant over the experiments: learning rate, the number of epochs in pre-training and training were 0.1, 50, 200/500 respectively. The average cross-entropy error for the train set was used for fine-tune phase where the mini-batch (contains 1000 random sample data) CG method with line search was applied. Batch size for pre-training was 100. In the fine-tune phase, the penalty term was 5×10^{-5}. Also, the network's topology and the number of hidden units were fixed for all experiments. In details, we use a 140-40-30-10 architecture in pre-training phase and a 140-40-30-10-30-40-140 architecture as the AE.

The results of the experiments is presented in the next sections of this chapter. Next section presents the applied evaluation metric of the thesis. The second section presents experiments on SKE where the generated summary is compared with both abstractive and extractive model summaries. The results of the BC3 corpus, where evaluations were compared with abstractive model summary, will be presented in the third section. In the fourth section, the generalization of the model is presented. The last section presents a discussion on the results, in general.

4.1 Evaluation Metric

The Evaluation of the produced summary is a challenging phase in a text summarization task. Questions such as is there any automatic metric to evaluate the correlation between generated summary and human generated one? If not is there any manual alternative? or even is there any need to have a golden standard human summary? are still open in this application domain (Louis and Nenkova 2013). One option is pyramid metric as an indirect manual evaluation. This technique has previously used for BC3. However, due to lack of resource we applied a fully automatic metric.

The ROUGE evaluation package has been commonly used as a fully automatic evaluation metric. This package was developed on the basis of n-gram similarity between the model summary (i.e. human generated) and the system summary (i.e. machine generated). Although accepting this metric across all domains has not been established, Dang et al. (2008) observed that among all ROUGE measures ROUGE-2 recall has the highest correlation with manual scores. In particular, ROUGE-2 recall has been applied for a deep structure in the evaluation of email summarization (Zajic et al. 2008, Genest et al. 2011). We provide the average ROUGE-2 recall for all experiments (see Appendix for tables showing Precision, Recall and f-score). Over the experiments, the confidence intervals is 95% and a jackknifing procedure was used for multi-annotation evaluation of ROUGE scores (Lin 2004). Also, to have more sample for training, we concatenated SKE, BC3 and the corpus developed by Molla et al. (2011)[1].

4.2 Evaluation on SKE

Among email datasets the Enron corpus is one of the most popular. This publicly available corpus has been commonly used for a wide variety of NLP purposes. Part of the corpus was manually annotated for text summarization in the University of British Columbia; however, it is not publicly available.

However, Loza et al. (2014) developed SKE and made it publicly available. This corpus was released for general-purpose summarization and keyword extraction for both extractive and abstractive summarization. SKE is a dataset that the majority of emails were selected from Enron email dataset. Other emails were provided by volunteers. The corpora consists of single emails and email threads where each email is manually annotated by two different annotators.

In SKE, there are 349 emails, from which 319 were selected from the Enron dataset and 30 were provided by volunteers. The 319 Enron emails were from 150 mailboxes and were categorized into private and corporate emails (i.e. emails were exchanged within work environment and between employees and employees or friends or their family). Both the private and corporate emails were divided into two parts: single email (with at least 10 lines) and email threads (with at least 3 emails). This categorization was also appears to the 30 emails provided by volunteers. This classification was presented in table 4-1.

[1] Molla et al. (2011) with 20939 samples has been used to provide additional data for both pre-training and training. The results are the same whether or not this corpus is included.

Group	Type	Count (from Enron email source)	Count (from Volunteers)
Single	Private	103	13
Single	Corporate	109	-
Thread	Private	45	17
Thread	Corporate	62	-

Table 4-1 the number of emails in each category over the dataset.

The corpus consists of a total of more than 100,000 words, 46,603 words after stemming and stop word removal process, and 7478 unique words. There is 6801 sentences (19.5 sentences per email) that are all uniquely identified. The average number of words per email is 303 and per sentence is 15.5. Figure 4-2 shows a randomly selected email. Quotation emails that are repetitions of text from earlier messages were removed by the developers.

For each email in the manual annotation process, each of two annotators created four types of annotation: an abstractive summary, an extractive summary, a set of key phrases and labeling each email as it is either private or corporate. Annotators were restricted to generate abstractive summaries with less than 450 characters (Loza et al. 2014). In practice, none of generated abstractive summaries are less than 33 words or more than 96 words. On average, the number of words for abstractive summary is about 73 words per email.

For extractive summaries, the annotators identified 5 different sentences (or lines) by selecting a subset of original sentences as the most important sentences of each email. This identification was indeed a sentence ranking. This reverse ranking was illustrated in figure 4-2. The process of identification was repeated for extracting 5 phrases. The developers ask annotators to extract at most 4 words for each key phrases and in practice they extracted 2.1 words per phrase, on average.

The dataset contains 7478 unique words that can provide a vocabulary for BOW representation. This was to be expected that the frequency of words decreases very rapidly with increasing the vocabulary size (figure 4-1). The frequency is more than 88, 54, 26, 9 and 1 for vocabulary size 60, 150, 374, 1000 and 3600 respectively.

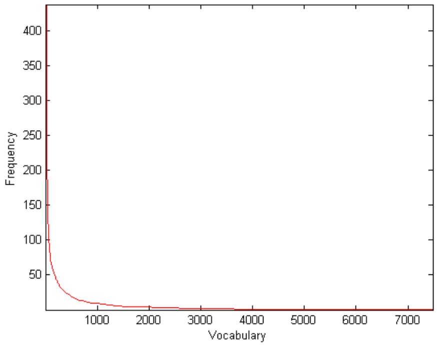
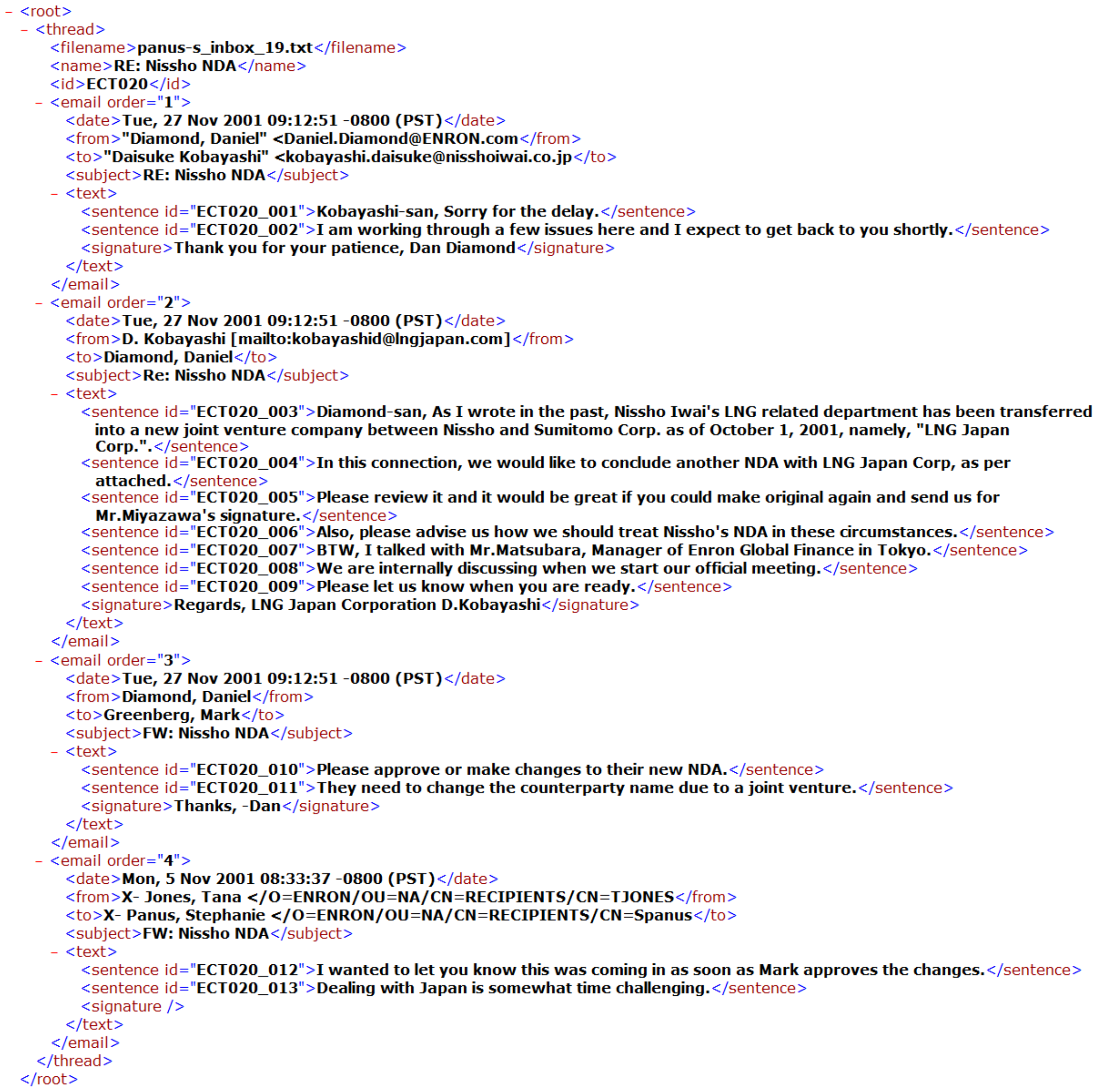


Figure 4-1 the frequency of the whole words.

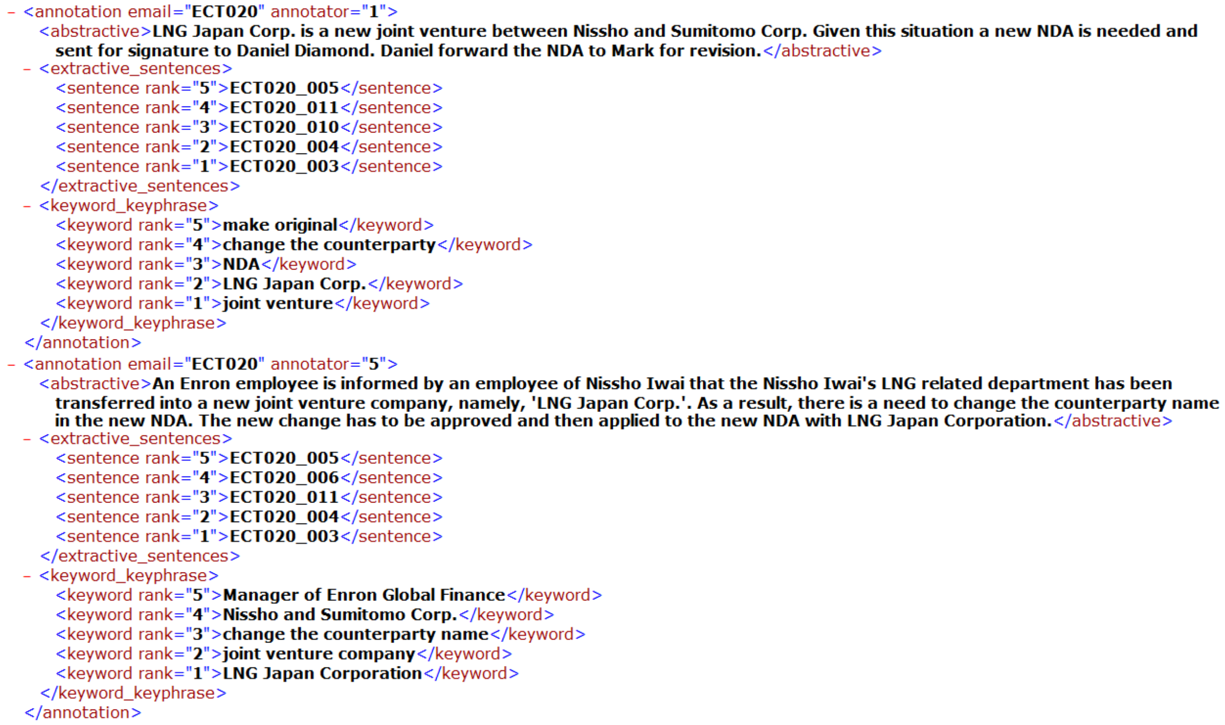


```
- <root>
  - <thread>
      <filename>panus-s_inbox_19.txt</filename>
      <name>RE: Nissho NDA</name>
      <id>ECT020</id>
    - <email order="1">
        <date>Tue, 27 Nov 2001 09:12:51 -0800 (PST)</date>
        <from>"Diamond, Daniel" <Daniel.Diamond@ENRON.com</from>
        <to>"Daisuke Kobayashi" <kobayashi.daisuke@nisshoiwai.co.jp</to>
        <subject>RE: Nissho NDA</subject>
      - <text>
          <sentence id="ECT020_001">Kobayashi-san, Sorry for the delay.</sentence>
          <sentence id="ECT020_002">I am working through a few issues here and I expect to get back to you shortly.</sentence>
          <signature>Thank you for your patience, Dan Diamond</signature>
        </text>
      </email>
    - <email order="2">
        <date>Tue, 27 Nov 2001 09:12:51 -0800 (PST)</date>
        <from>D. Kobayashi [mailto:kobayashid@lngjapan.com]</from>
        <to>Diamond, Daniel</to>
        <subject>Re: Nissho NDA</subject>
      - <text>
          <sentence id="ECT020_003">Diamond-san, As I wrote in the past, Nissho Iwai's LNG related department has been transferred
            into a new joint venture company between Nissho and Sumitomo Corp. as of October 1, 2001, namely, "LNG Japan
            Corp.".</sentence>
          <sentence id="ECT020_004">In this connection, we would like to conclude another NDA with LNG Japan Corp, as per
            attached.</sentence>
          <sentence id="ECT020_005">Please review it and it would be great if you could make original again and send us for
            Mr.Miyazawa's signature.</sentence>
          <sentence id="ECT020_006">Also, please advise us how we should treat Nissho's NDA in these circumstances.</sentence>
          <sentence id="ECT020_007">BTW, I talked with Mr.Matsubara, Manager of Enron Global Finance in Tokyo.</sentence>
          <sentence id="ECT020_008">We are internally discussing when we start our official meeting.</sentence>
          <sentence id="ECT020_009">Please let us know when you are ready.</sentence>
          <signature>Regards, LNG Japan Corporation D.Kobayashi</signature>
        </text>
      </email>
    - <email order="3">
        <date>Tue, 27 Nov 2001 09:12:51 -0800 (PST)</date>
        <from>Diamond, Daniel</from>
        <to>Greenberg, Mark</to>
        <subject>FW: Nissho NDA</subject>
      - <text>
          <sentence id="ECT020_010">Please approve or make changes to their new NDA.</sentence>
          <sentence id="ECT020_011">They need to change the counterparty name due to a joint venture.</sentence>
          <signature>Thanks, -Dan</signature>
        </text>
      </email>
    - <email order="4">
        <date>Mon, 5 Nov 2001 08:33:37 -0800 (PST)</date>
        <from>X- Jones, Tana </O=ENRON/OU=NA/CN=RECIPIENTS/CN=TJONES</from>
        <to>X- Panus, Stephanie </O=ENRON/OU=NA/CN=RECIPIENTS/CN=Spanus</to>
        <subject>FW: Nissho NDA</subject>
      - <text>
          <sentence id="ECT020_012">I wanted to let you know this was coming in as soon as Mark approves the changes.</sentence>
          <sentence id="ECT020_013">Dealing with Japan is somewhat time challenging.</sentence>
          <signature />
        </text>
      </email>
    </thread>
  </root>
```

Figure 4-2 A randomly selected sample email thread from corporate section (ECT020.xml). The subject of this email is "Nissho NDA" and it consists of 4 emails and 13 sentences.

SKE developers provide some information regarding agreement of annotators, for both summary and key phrase extraction:

> Considering one annotator as the ground truth, and another as the "system", the agreement on the keyword extraction task was 25.33% precision, 25.33% recall, 25.33% F-score, and 14.50% Jaccard similarity. For sentence extraction task the values were 51.33% precision, 51.33% recall, 51.33% F-score. Precision, recall, and F-score values are the same because both annotators annotated the same number of key phrases and sentences per document. The validation of the private/corporate classification during the annotation stage shows a 95% concordance for both annotators, and an inter-annotator agreement of 88%. (Loza et al. 2014)



```
- <annotation email="ECT020" annotator="1">
    <abstractive>LNG Japan Corp. is a new joint venture between Nissho and Sumitomo Corp. Given this situation a new NDA is needed and
      sent for signature to Daniel Diamond. Daniel forward the NDA to Mark for revision.</abstractive>
  - <extractive_sentences>
      <sentence rank="5">ECT020_005</sentence>
      <sentence rank="4">ECT020_011</sentence>
      <sentence rank="3">ECT020_010</sentence>
      <sentence rank="2">ECT020_004</sentence>
      <sentence rank="1">ECT020_003</sentence>
    </extractive_sentences>
  - <keyword_keyphrase>
      <keyword rank="5">make original</keyword>
      <keyword rank="4">change the counterparty</keyword>
      <keyword rank="3">NDA</keyword>
      <keyword rank="2">LNG Japan Corp.</keyword>
      <keyword rank="1">joint venture</keyword>
    </keyword_keyphrase>
  </annotation>
- <annotation email="ECT020" annotator="5">
    <abstractive>An Enron employee is informed by an employee of Nissho Iwai that the Nissho Iwai's LNG related department has been
      transferred into a new joint venture company, namely, 'LNG Japan Corp.'. As a result, there is a need to change the counterparty name
      in the new NDA. The new change has to be approved and then applied to the new NDA with LNG Japan Corporation.</abstractive>
  - <extractive_sentences>
      <sentence rank="5">ECT020_005</sentence>
      <sentence rank="4">ECT020_006</sentence>
      <sentence rank="3">ECT020_011</sentence>
      <sentence rank="2">ECT020_004</sentence>
      <sentence rank="1">ECT020_003</sentence>
    </extractive_sentences>
  - <keyword_keyphrase>
      <keyword rank="5">Manager of Enron Global Finance</keyword>
      <keyword rank="4">Nissho and Sumitomo Corp.</keyword>
      <keyword rank="3">change the counterparty name</keyword>
      <keyword rank="2">joint venture company</keyword>
      <keyword rank="1">LNG Japan Corporation</keyword>
    </keyword_keyphrase>
  </annotation>
```

Figure 4-3 the abstractive summaries, 5 selected sentences and key-phrases of the sample email presented in figure 4-1.

4.2.1 Experiment Setup

To analyze the structure, we perform three different threads of experiments on SKE. First, we generate summaries based on the subject of each email. Although, in the real world, the subject of emails commonly contains informative words, one problem in this approach was empty subjects of SKE. To solve the problem we excluded emails with empty subjects. After this removal, there was 289 emails out of 349 emails. On average, each email of this data set has 3.1 words per subject.

In the second experiment, we generate summaries using the annotated key phrases as queries (key-phrase-oriented summarization). In the third subsection, this key-phrase-oriented summaries will be compared with abstractive summaries generated by annotators. It is true that our summarization is extractive, in the third experiment the extracted summaries will be analogue with human generate abstractive summary as a golden standard. As all emails in the corpus have been annotated with keyword phrases, this experiment was performed on the whole dataset (i.e. 349 emails).

Since 5 sentences were selected by annotators and this number is fixed for all emails, the system summaries also contain 5 sentences in all experiments of this section. When it comes to implementation, we merge all different emails, both private and corporate.

4.2.2 Results

To produce comparable results, we chose the same settings for the three experiments that includes the size of vocabulary, the number of experiments and hidden units/layers of the network. Also, all additive noise had the same variance and were generated randomly for each experiment.

4.2.2.1. Subject-oriented Summarization

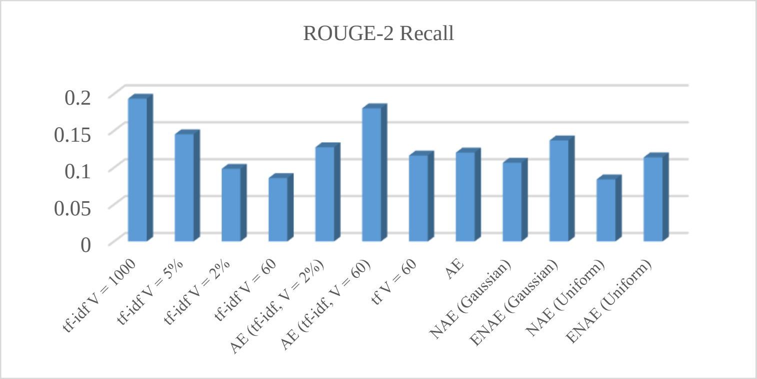
As a convention, a writer of an email tries to choose the most important words and concepts for the subject of the email. In this experiment, the subject is used as a query to extract a summary related to the subject. In this way, the AE will change the input representation (i.e. word space) into a low-dimensional feature space (i.e. concept space). More precisely, AE has been used to map the representation from BOW into a concept space (i.e. codes generated in the bottleneck layer).

In this experiment, SKE (289 emails) contains 6423 unique words and we constructed *tf-idf* vectors based on the 1000, 320 (5% of the whole vocabulary), 128 (2% of the whole vocabulary), and 60 most frequently occurring words. This corresponds to words with a minimum frequency of at least 7, 24, 48 and 76 for vocabulary size 1000, 320, 128 and 60 respectively. For *tf* representation the size of vocabulary is 60 and to have a comparable result we present $v = 60$ for *tf-idf* as well.

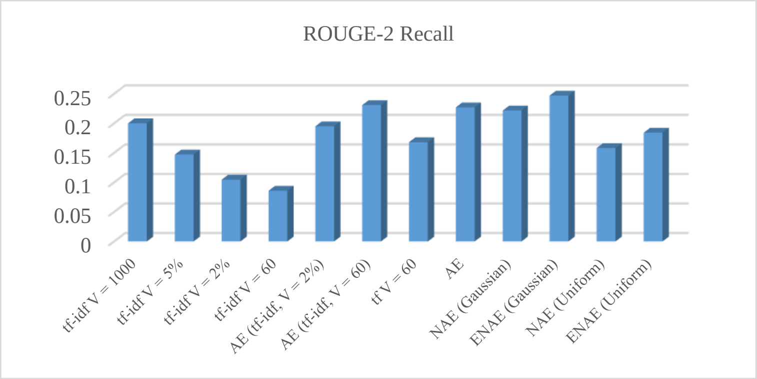
Figure 4-4 illustrates the ROUGE-2 recall of the *tf-idf* baselines and the AE model with various input representations. This is hardly surprising that increasing the number of sentences (i.e. summaries with 1, 2, 3, 4, 5 sentences) results in a slight improvement of the recall for *tf-idf*. This increment is to some extend constant for *tf-idf* with different size of vocabulary. This trend is somewhat the same when *tf-idf* representation has been used as the input of AE. However, *tf-idf* with longer vocabulary size provides less effective representation for AE. This may come from the sparsity. That is, most of the *tf-idf* matrix contains zero and increasing the size of vocabulary increases the ratio of zero element to non-zero.

The main thing to note is that AE (*tf*) (i.e. *tf* as the input of AE) is always much better than *tf*. The difference between ROUGE scores even becomes larger when the number of sentences of summaries is increased. It shows that AE can enhance the representation and provide more discriminative features.

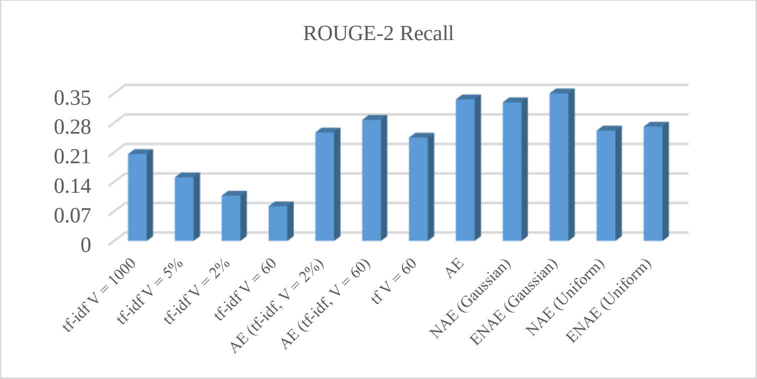
Increasing the number of sentences significantly improves the ROUGE score of AE with all variants of representation; however, the scale of this improvement is not the same for all. In general, the ROUGE score of AE (*tf*), Noisy Auto-Encoder (NAE) and Ensemble Noisy Auto-Encoder (ENAE), that are perturbed with either Gaussian or Uniform noise, is itself noisy (i.e. having no deterministic order) for summaries with only one sentence. This trend is not the same when the number of extracted sentences increases. The ROUGE score of AE with all variants of input representation is closer to each other for summaries with 5 sentences.


ROUGE-2 Recall
0.2
0.15
0.1
0.05
0
tf-idf V = 1000
tf-idf V = 5%
tf-idf V = 2%
tf-idf V = 60
AE (tf-idf, V = 2%)
AE (tf-idf, V = 60)
tf V = 60
AE
NAE (Gaussian)
ENAE (Gaussian)
NAE (Uniform)
ENAE (Uniform)

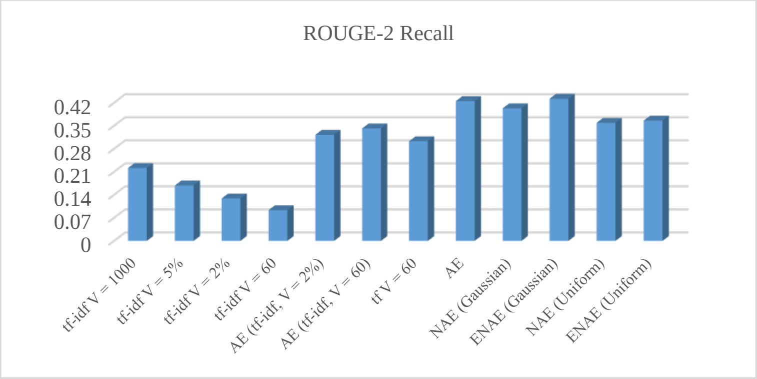
(a)


ROUGE-2 Recall
0.25
0.2
0.15
0.1
0.05
0
tf-idf V = 1000
tf-idf V = 5%
tf-idf V = 2%
tf-idf V = 60
AE (tf-idf, V = 2%)
AE (tf-idf, V = 60)
tf V = 60
AE
NAE (Gaussian)
ENAE (Gaussian)
NAE (Uniform)
ENAE (Uniform)

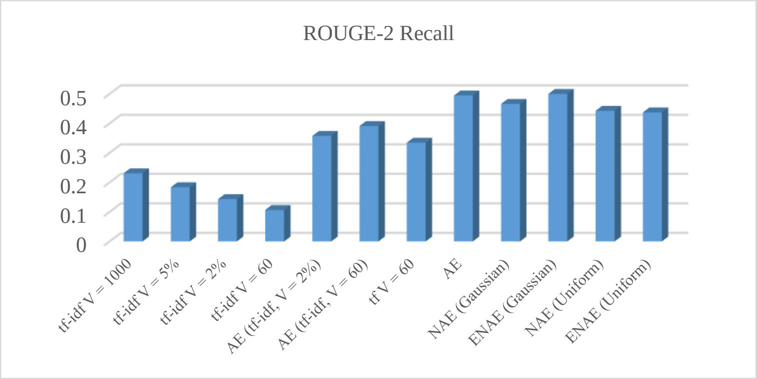
(b)


ROUGE-2 Recall
0.35
0.28
0.21
0.14
0.07
0
tf-idf V = 1000
tf-idf V = 5%
tf-idf V = 2%
tf-idf V = 60
AE (tf-idf, V = 2%)
AE (tf-idf, V = 60)
tf V = 60
AE
NAE (Gaussian)
ENAE (Gaussian)
NAE (Uniform)
ENAE (Uniform)

(c)


ROUGE-2 Recall
0.42
0.35
0.28
0.21
0.14
0.07
0
tf-idf V = 1000
tf-idf V = 5%
tf-idf V = 2%
tf-idf V = 60
AE (tf-idf, V = 2%)
AE (tf-idf, V = 60)
tf V = 60
AE
NAE (Gaussian)
ENAE (Gaussian)
NAE (Uniform)
ENAE (Uniform)

(d)



(e)

Figure 4-4 ROUGE-2 recall score for *tf-idf*, *tf* and AE with different input representations. (a), (b), (c), (d) and (e) are ROUGE scores when the summaries contains 1, 2, 3, 4 and 5 sentences respectively.

		n = 1	n = 2	n = 3	n = 4	n = 5
tf-idf V = 1000	(**Recall**)	0.1936	0.2002	0.2062	0.2217	0.2312
	(Precision)	0.1984	0.2030	0.2098	0.2188	0.2254
tf-idf V = 5%	(**Recall**)	0.1453	0.1473	0.1509	0.1688	0.1838
	(Precision)	0.1482	0.1500	0.1508	0.1641	0.1764
tf-idf V = 2%	(**Recall**)	0.0985	0.1048	0.1074	0.1291	0.1435
	(Precision)	0.1014	0.1089	0.1095	0.1229	0.1363
tf-idf V = 60	(**Recall**)	0.0859	0.0860	0.0823	0.0935	0.1068
	(Precision)	0.0886	0.0901	0.0838	0.0907	0.1041
AE (*tf-idf*, *V* = 2%)	(**Recall**)	0.1277	0.1951	0.2572	0.3227	0.3580
	(Precision)	0.1354	0.2265	0.2980	0.3724	0.4134
AE (*tf-idf*, *V* = 60)	(**Recall**)	0.1805	0.2310	0.2871	0.3422	0.3913
	(Precision)	0.1876	0.2567	0.3196	0.3853	0.4429
tf V = 60	(**Recall**)	0.1165	0.1682	0.2452	0.3032	0.3349
	(Precision)	0.1213	0.1827	0.2498	0.2949	0.3208
AE (*tf*)	(**Recall**)	0.1205	0.2272	0.3360	0.4251	0.4948
	(Precision)	0.1286	0.2409	0.3565	0.4457	0.5270
NAE (Gaussian)	(**Recall**)	0.1067	0.2219	0.3289	0.4033	0.4664
	(Precision)	0.1119	0.2368	0.3518	0.4300	0.5029
ENAE (Gaussian)	(**Recall**)	0.1370	0.2471	0.3510	0.4325	0.5031
	(Precision)	0.1385	0.2569	0.3653	0.4498	0.5264
NAE (Uniform)	(**Recall**)	0.0841	0.1583	0.2618	0.3592	0.4428
	(Precision)	0.0912	0.1961	0.3106	0.4121	0.5052
ENAE (Uniform)	(**Recall**)	0.1140	0.1843	0.2711	0.3659	0.4377
	(Precision)	0.1177	0.2077	0.3064	0.4035	0.4867

Table 4-2 ROUGE-2 recall of *tf-idf*, *tf* and models versus the number of selected sentences. n is the number of sentences for a summary.

Table 4.2 more clearly presents the ROUGE scores. It shows more relevant sentences can be extracted using ENAE (Gaussian) and AE (*tf*) compared to NAE (with either Gaussian or Uniform noise) and ENAE with voted from Uniform noise runs. For summaries with one sentences the best approach is *tf-idf*; however, for summaries with more sentences AE preforms much better. This may show that for subject-based summaries mapping term space to concept space is not quite accurate for every single vector, but the mapping is much useful on average (i.e. the summaries with more sentences). The highest performance of *tf-idf*, where summaries contains 5 sentence, is still less than half of ENAE (Gaussian).

To have a more exact evaluation, we analyze the sample email presented in figure 4-1. As presented in figure 4-5, from 5 extracted sentences, 3 and 4 sentences were also extracted by annotator 1 and 2 respectively. It is true that sentence number 5 has not been ranked among the first 5 sentences by the system, but this sentence has been ranked at the last priority by both annotators. In general, only sentence number 8 is not ranked by annotators. In this example, sentences number 3, 8, 4, 6 and 11 are ranked first to fifth by the system. The ranking is sentences number 3, 4, 10, 11 and 5 by first annotator and 3, 4, 11, 6 and 5 by second annotator.

Subject of email: Nissho NDA

System summary (ENAE):

03 - Diamond-san, As I wrote in the past, Nissho Iwai's LNG related department has been transferred into a new joint venture company between Nissho and Sumitomo Corp. as of October 1, 2001, namely, "LNG Japan Corp.".
08 - We are internally discussing when we start our official meeting.
04 - In this connection, we would like to conclude another NDA with LNG Japan Corp, as per attached.
06 - Also, please advise us how we should treat Nissho's NDA in these circumstances.
11 - They need to change the counterparty name due to a joint venture.

First Annotator

Extractive summary:

03 - Diamond-san, As I wrote in the past, Nissho Iwai's LNG related department has been transferred into a new joint venture company between Nissho and Sumitomo Corp. as of October 1, 2001, namely, "LNG Japan Corp.".
04 - In this connection, we would like to conclude another NDA with LNG Japan Corp, as per attached.
10 - Please approve or make changes to their new NDA.
11 - They need to change the counterparty name due to a joint venture.
05 - Please review it and it would be great if you could make original again and send us for Mr.Miyazawa's signature.

Second Annotator

Extractive summary:

03 - Diamond-san, As I wrote in the past, Nissho Iwai's LNG related department has been transferred into a new joint venture company between Nissho and Sumitomo Corp. as of October 1, 2001, namely, "LNG Japan Corp.".
04 - In this connection, we would like to conclude another NDA with LNG Japan Corp, as per attached.
11 - They need to change the counterparty name due to a joint venture.
06 - Also, please advise us how we should treat Nissho's NDA in these circumstances.
05 - Please review it and it would be great if you could make original again and send us for Mr.Miyazawa's signature.

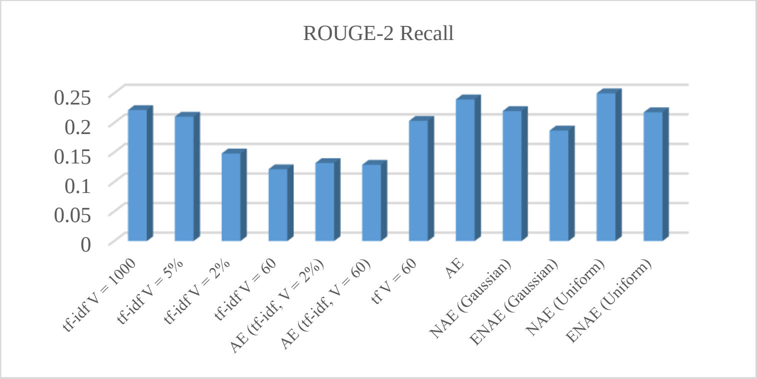
Figure 4-5 the extracted summary using ENAE (Gaussian) and the extractive summaries of both annotators from the sample email thread presented in figure 4-1.

4.2.2.2. Key-phrase-oriented Summarization

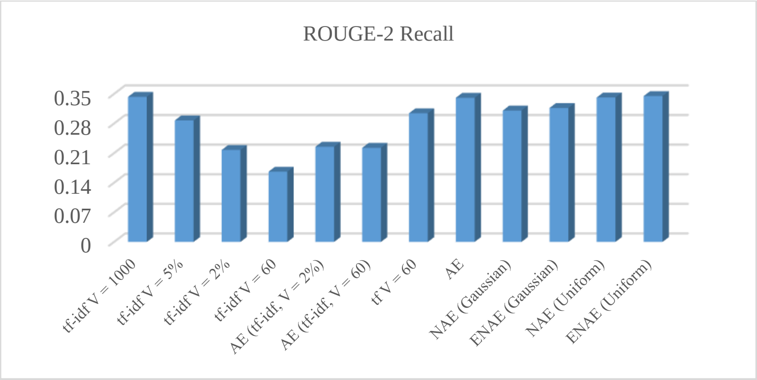
Text Summarization based on key phrase extraction has been extensively studied (Qazvinian et al. 2010, Zhao et al. 2011, Bhaskar 2013). Key phrase extraction itself requires complicated technique and this thesis does not cover it; however, SKE provides the key phrase of each email and we explore this application domain using the AE.

In this experiment, the five manually extracted key phrases are a query. We constructed *tf-idf* vectors based on the 1000, 374 (5% of the whole vocabulary), 150 (2% of the whole vocabulary) and 60 most frequently occurring words. This corresponds to words with a minimum frequency of at least 9, 26, 54 and 88 for vocabulary size 1000, 374, 150 and 60 respectively. For *tf* representation the size of vocabulary is 60 and to have a comparable result we present $v = 60$ for *tf-idf* as well.

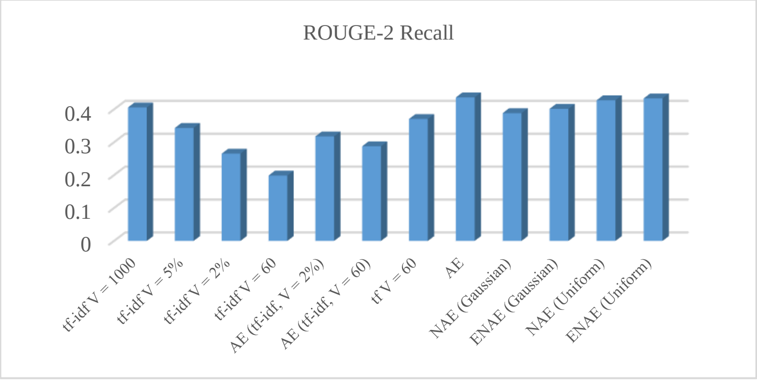
Figure 4-6 shows the ROUGE-2 recall of the *tf-idf* baselines and the AE model with various input representations. It was expected that using carefully extracted key phrases can help improve the performance of the summarization. Although this improvement can be seen for all techniques, the rate of this improvement is not the same for each technique.



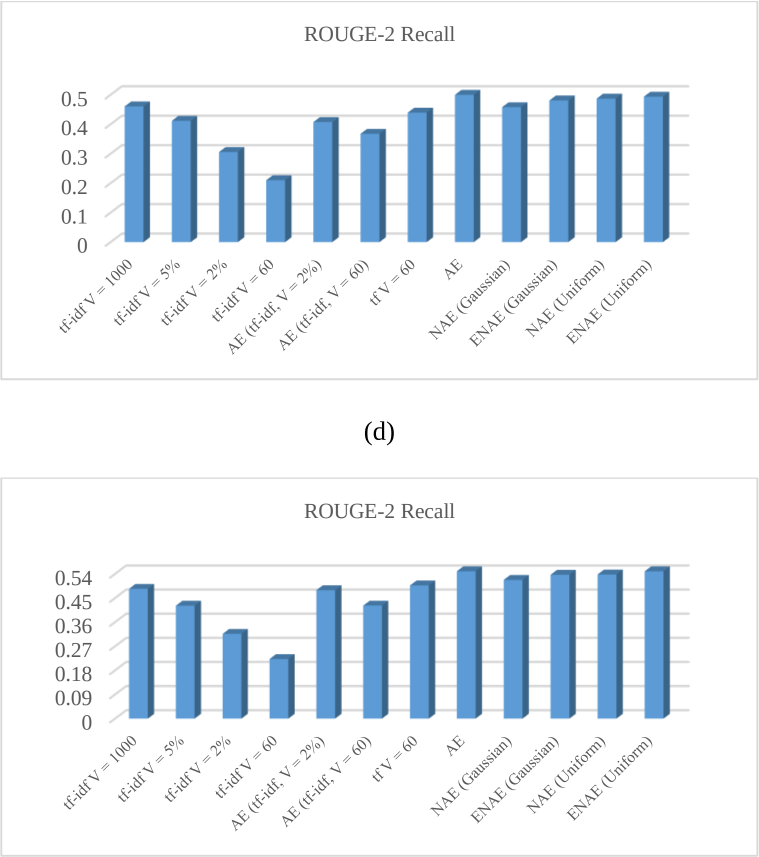
(a)



(b)



(c)



(d)

(e)

Figure 4-6 ROUGE-2 recall score for *tf-idf*, *tf* and AE with different input representations. (a), (b), (c), (d) and (e) are ROUGE scores when the summaries contains 1, 2, 3, 4 and 5 sentences respectively.

With only one sentence in summaries, *tf-idf* improves the results compared to its counterpart in subject-oriented summarization, but not significantly. Using *tf-idf*, increasing the number of sentences enhances the summarization quality and the ROUGE score for summaries with 5 sentences is more than 2 times of its counterpart in subject-oriented summarization.

Key phrases contribute the AE (*tf-idf*) to improve the result but the trend is opposite to the subject-oriented summarization. More clearly, AE (*tf-idf*) with vocabulary size 150 preforms better than 60 words. The reason may lie in the importance of the query. In contrast with subject of email, key phrases provide more important information and it can make more influence than sparsity increment.

The performance of AE (*tf*) is better than *tf*. A key difference between subject-oriented and key-phrase-oriented summarization for SKE corpus is when the length of extracted summaries is one sentence. In $n = 1$, the ROUGE score of the AE with various input representations is higher than *tf-idf* and AE (*tf-idf*). The reason may come from the centrality of the query. Calculating the distance of a query and a sentence (i.e. considering the query in the center of the given document) requires more accurate place of the query in the concept space. Also, even if we assume that the subject of an email is as important as key phrases, the average length of each subject is about 3.1 words per email while this length is 10.5 (2.1 words per phrase) for a key-phrase-based query. The differences help queries to be placed in a more appropriate position in the term space and thereby concept space.

In summaries with more sentences, AE (*tf*) mostly keeps its superior and for five sentence summaries AE with various *tf* representations are better than *tf-idf* and *tf*. Table 4.3 shows that AE (*tf*) and ENAE (uniform) provide better results in comparison with other techniques.

tf-idf with the vocabulary size 150 and 60 words cannot provide good performance. However, transferring *tf-idf* representation into the concept space using AE significantly enhances the quality of extracted summaries. For example, the highest ROUGE score of *tf-idf* with $v = 150$ and $v = 60$ is 0.3166 and 0.2224 respectively; whereas this score is 0.4795 and 0.4220 for AE (*tf-idf*) with the vocabulary size 150 and 60 respectively.

		n = 1	**n = 2**	**n = 3**	**n = 4**	**n = 5**
tf-idf V = 1000	(**Recall**)	0.2217	0.3432	0.4059	0.4607	0.4845
	(Precision)	0.2252	0.3414	0.4101	0.4613	0.4817
tf-idf V = 5%	(**Recall**)	0.2105	0.2874	0.3437	0.4118	0.4217
	(Precision)	0.2124	0.2929	0.3478	0.4040	0.4120
tf-idf V = 2%	(**Recall**)	0.1482	0.2176	0.2655	0.3060	0.3166
	(Precision)	0.1486	0.2160	0.2643	0.2942	0.3036
tf-idf V = 60	(**Recall**)	0.1215	0.1661	0.1988	0.2103	0.2224
	(Precision)	0.1242	0.1616	0.1895	0.2000	0.2119
AE (*tf-idf*, *V* = 2%)	(**Recall**)	0.1321	0.2251	0.3179	0.4077	0.4795
	(Precision)	0.1372	0.2388	0.3388	0.4324	0.5139
AE (*tf-idf*, *V* = 60)	(**Recall**)	0.1291	0.2226	0.2879	0.3678	0.4220
	(Precision)	0.1337	0.2412	0.3303	0.4098	0.4762
tf V = 60	(**Recall**)	0.2034	0.3044	0.3708	0.4395	0.4972
	(Precision)	0.2060	0.3064	0.3760	0.4533	0.5134
AE (*tf*)	(**Recall**)	0.2397	0.3410	0.4365	0.5035	0.5657
	(Precision)	0.2339	0.3225	0.4069	0.4785	0.5556
NAE (Gaussian)	(**Recall**)	0.2200	0.3107	0.3887	0.4581	0.5179
	(Precision)	0.2209	0.3105	0.3966	0.4643	0.5359
ENAE (Gaussian)	(**Recall**)	0.1870	0.3168	0.4017	0.4809	0.5370
	(Precision)	0.1869	0.3115	0.3974	0.4742	0.5406
NAE (Uniform)	(**Recall**)	0.2547	0.3416	0.4277	0.4872	0.5380
	(Precision)	0.2514	0.3209	0.4055	0.4685	0.5228
ENAE (Uniform)	(**Recall**)	0.2179	0.3450	0.4334	0.4938	0.5504
	(Precision)	0.2149	0.3298	0.4100	0.4774	0.5406

Table 4-3 ROUGE-2 recall of *tf-idf*, *tf* and models versus the number of selected sentences. n is the number of sentences for a summary.

Figure 4-7 explores the case study presented in 4.1. The intersection between the system summary and the model summary (2 annotators) is the sentence number 10, 3, 4 and 6. The key change from the subject-oriented summarization is the order of the ranking and the inclusion of the sentence number 12. Indeed, if we calculate precision and recall score of this sample email, the extraction performance is the same as the subject-oriented summarization, but the indexes cannot present the accuracy of ranking order for truly extracted sentences. In the sample subject-oriented summary, mistaken extracted sentence had high rank whereas with key phrases sentence number 12 was ranked lower. We will see in the next experiment that sentence number 12 is not a fault extraction.

System summary (ENAE):

10 - Please approve or make changes to their new NDA.
03 - Diamond-san, As I wrote in the past, Nissho Iwai's LNG related department has been transferred into a new joint venture company between Nissho and Sumitomo Corp. as of October 1, 2001, namely, "LNG Japan Corp.".
04 - In this connection, we would like to conclude another NDA with LNG Japan Corp, as per attached.
12 - I wanted to let you know this was coming in as soon as Mark approves the changes.
06 - Also, please advise us how we should treat Nissho's NDA in these circumstances.

First Annotator

-----------------------

Extractive summary:

03 - Diamond-san, As I wrote in the past, Nissho Iwai's LNG related department has been transferred into a new joint venture company between Nissho and Sumitomo Corp. as of October 1, 2001, namely, "LNG Japan Corp.".
04 - In this connection, we would like to conclude another NDA with LNG Japan Corp, as per attached.
10 - Please approve or make changes to their new NDA.
11 - They need to change the counterparty name due to a joint venture.
05 - Please review it and it would be great if you could make original again and send us for Mr.Miyazawa's signature.

Key phrases:

1 - Joint venture
2 - LNG Japan Corp.
3 - NDA
4 - Change the counterparty
5 - Make original

Second Annotator

---------------------------

Extractive summary:

03 - Diamond-san, As I wrote in the past, Nissho Iwai's LNG related department has been transferred into a new joint venture company between Nissho and Sumitomo Corp. as of October 1, 2001, namely, "LNG Japan Corp.".
04 - In this connection, we would like to conclude another NDA with LNG Japan Corp, as per attached.
11 - They need to change the counterparty name due to a joint venture.
06 - Also, please advise us how we should treat Nissho's NDA in these circumstances.
05 - Please review it and it would be great if you could make original again and send us for Mr.Miyazawa's signature.

Key phrases:

1 - LNG Japan Corporation
2 - Joint venture company
3 - Change the counterparty name
4 - Nissho and Sumitomo Corp.
5 - Manager of Enron Global Finance

Figure 4-7 The extracted summary based on key phrases using ENAE (Gaussian), the extractive summaries and key phrases of both annotators from the sample email thread presented in figure 4-1.

4.2.2.3. Key-phrase-oriented Summarization Versus Abstractive Summary

In addition to extracted sentences by annotators, SKE has abstractive summaries. In this section, the system summary is extracted sentences whereas the model summary is abstractive summaries (generated by human). The abstractive summaries can be seen as the golden standard for a system and it can be fruitful to analyze the results of techniques with this golden standard. The average number of words per email is about 73 for abstractive summaries.

Table 4.4 present the ROUGE-2 recall of the extracted sentences compared with abstractive summaries. It was expected that the score (i.e. recall) decrease to small values because the extracted sentences are being compared with the abstractive summaries generated by human. However, the trend is the same when the extracted sentences are compared with extractive summaries. This experiment shows that AE (*tf*) can perform better than *tf-idf* and *tf*.

	Recall	Precision
tf-idf V = 1000	0.2120	0.1552
tf-idf V = 5%	0.1848	0.1310
tf-idf V = 2%	0.1424	0.0966
tf-idf V = 60	0.0942	0.0629
AE (*tf-idf*, *V* = 2%)	0.2115	0.1694
AE (*tf-idf*, *V* = 60)	0.1875	0.1632
tf V = 60	0.2137	0.1618
AE (*tf*)	0.2319	0.1679
NAE (Gaussian)	0.2110	0.1647
ENAE (Gaussian)	0.2255	0.1676
NAE (Uniform)	0.2210	0.1600
ENAE (Uniform)	0.2299	0.1675

Table 4-4 ROUGE-2 recall of *tf-idf*, *tf* and models.

To show whether the incorrect extracted sentences are related to a golden standard or not, the abstractive summaries can be an appropriate benchmark. Figure 4-8 presents the abstractive summaries by annotators and the result of ENAE (Gaussian). Taking the sentence number 12 into consideration, we can find out that although this sentences was not chosen for extractive summaries by annotators, it contains informative information that both annotators has been informed in their abstractive summaries.

System summary (ENAE):

10 - Please approve or make changes to their new NDA.
03 - Diamond-san, As I wrote in the past, Nissho Iwai's LNG related department has been transferred into a new joint venture company between Nissho and Sumitomo Corp. as of October 1, 2001, namely, "LNG Japan Corp.".
04 - In this connection, we would like to conclude another NDA with LNG Japan Corp, as per attached.
12 - I wanted to let you know this was coming in as soon as Mark approves the changes.
06 - Also, please advise us how we should treat Nissho's NDA in these circumstances.

First Annotator

-----------------------

Abstractive summary:

- LNG Japan Corp. is a new joint venture between Nissho and Sumitomo Corp. Given this situation a new NDA is needed and sent for signature to Daniel Diamond. Daniel forward the NDA to Mark for revision.

Key phrases

1 - Joint venture
2 - LNG Japan Corp.
3 - NDA
4 - Change the counterparty
5 - Make original

Second Annotator

Abstractive summary:

- An Enron employee is informed by an employee of Nissho Iwai that the Nissho Iwai's LNG related department has been transferred into a new joint venture company, namely, 'LNG Japan Corp.'. As a result, there is a need to change the counterparty name in the new NDA. The new change has to be approved and then applied to the new NDA with LNG Japan Corporation.

Key phrases

1 - LNG Japan Corporation
2 - Joint venture company
3 - Change the counterparty name
4 - Nissho and Sumitomo Corp.
5 - Manager of Enron Global Finance

Figure 4-8 The extracted summary based on key phrases using ENAE (Gaussian), the abstractive summaries and extracted key phrases by annotators from the sample email thread presented in figure 4-1.

4.3 Evaluation on BC3

It can be more helpful when the model provides summaries from other annotated corpus. To do so, we chose BC3 dataset collected from World Wide Web Consortium[1] email corpus. W3C contains of more than 50,000 email threads taken from mailing lists. Emails from mailing lists mostly are not as informal and personal as a mailbox.

BC3 is a publicly available dataset specifically developed for summarization. The dataset contains of 40 email threads (3222 sentences) each of which on average has less than 11 emails and less than 6 participants. In the annotation process, 10 annotators were involved and each email thread was annotated by 3 different annotators. The dataset has been annotated for both extractive and abstractive summarization (with a 250 word limit) and labeled with speech act, meta sentences and subjectivity (Ulrich et al. 2008). We only use the abstractive summaries.

The raw BC3 contains of 3222 sentences (lines in xml format), but in practice, the dataset needs to be cleaned and organized. We removed repetitions of text from early massages for each thread. Also, we eliminated signatures, sign offs and senders' name. This left 736 sentences (18.4 sentence per email thread) and more than 15,000 words. We further preprocessed the data by stemming and removing common stop words. After this step, there were 7101 words. The number of unique words was 2306. The average number of words per email thread and per sentence was 380 and 21 respectively.

Figure 4-9 shows a typical email thread for BC3. The subject of this email thread is "Next face to face meeting" and the email thread includes 6 emails. This email has been cleaned and organized. One example of this organization can be seen in the first email, sentence 1 (with sentence number 1-1) where "Dear all" counts as a sentence and "Hoping that we will …" counts as a separate sentence, in the original BC3 corpus. Although "Dear all" is not informative, we would not lose it. We reorganized it to one sentence "Dear all, Hoping that we will …". This reorganization is because we want to have similar settings as SKE.

In the BC3 corpus, there was no limitation for annotators to identify important sentences. In practice, for extractive summarization, annotators extracted at least 4, at most 34 and on average 14 sentences per email thread. However, for abstractive summarization, annotators were restricted to generate less than 250 words for each email thread and on average the generated summaries has 122 words per email thread.

[1] http://research.microsoft.com/en-us/um/people/nickcr/w3c-summary.html

Subject of email: Next face to face meeting

Email Thread

1-1 Dear all, Hoping that we will have a last call review shortly (see my next email) we are planning a face to face meeting at which we can resolve issues raised, and generally tidy the document up (write up lots more good techniques).
1-2 At the moment we are talking to a potential host in the Boston area about a meeting on thursday and friday 7-8 October.
1-3 More details as they come to hand.
2-1 Folks, Please note that this meeting would be on at the same time as the ATIA meeting in Orlando, Florida.
2-2 If the time is impossible for anyone please let us know as soon as possible at the moment the next closest possible times are a week earlier (which will cut into the time available for last call review) or ten days later (which will possibly push our schedule back further).
3-1 Hello, I have other meetings tentatively scheduled for 6 and 7 October in Boston.
3-2 I would probably be able to attend starting the 7th.
4-1 I believe that I will be in Rome on the 6th and 7th.
5-1 Dear working group participants, As discussed, the meeting will indeed be held on thursday 7 and friday 8 October.
5-2 I am pleased to announce that Allaire Corp will be our hosts, and look forward to a productive two days, and some time to say hi. The nitty-gritty.
5-3 Meeting registration is open from now until 1 October.
5-4 register via the meeting page at http://www.w3.org/WAI/AU/f2f-oct99 which also provides details on location, accommodation, etc.
5-5 There are a limited number of rooms available in the Royal Sonesta Hotel (which is the meeting venue) at a discounted rate in order to qualify for this rate reservations and meeting registration must be completed by September 10.
5-6 Since October is the peak season for Boston hotels, rooms are expensive and hotels book out early, so you are advised to make your plans and reservations as soon as possible.
5-7 If you have any questions or special requirements please contact me as early as possible.
6-1 Most important if you haven't registered yet, don't forget.
6-2 http://www.w3.org/WAI/AU/f2f-oct99 for the details.
6-3 Given that there are a small number of us, I thought I would ask what people want to eat for lunch.
6-4 The default choices are roast turkey breast with vegetables (without turkey if you are a vegan), and pasta with vegetable rataouille.
6-5 Possible substitutions: baked Scrod (fish) chicken caesar salad hommus roll-up sandwich with salad salad with crab cakes churrasco flank steak with chimichurri and onions.
6-6 I will do this in the following way: If you cannot eat one of these things please make that very clear.
6-7 It will take at least two votes to change something, and will have to be compatible with what people can't do.
6-8 The deadline is 3pm (boston time) Monday.
6-9 No response will not be considered a vote for the default it will be considered a vote for "whatever gets chosen" Sorry to bore you with administrivia, but I figure that having food that people like helps the meeting go well.

Figure 4-9 a sample email thread from BC3 (list number: 058-13714734).

It was expected that the frequency of words decreases very rapidly with increasing the vocabulary size (figure 4-10). This frequency is more than 16, 10 and 1 for vocabulary size 60, 120 and 1000 respectively.

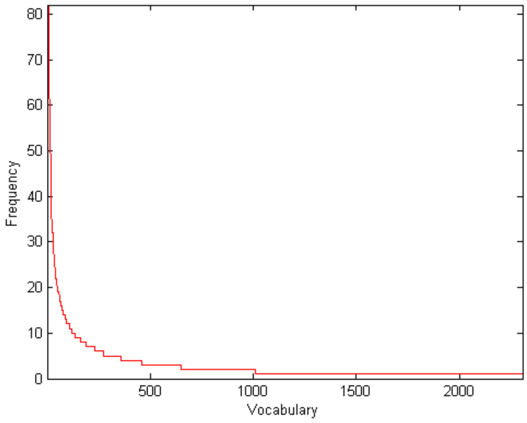


Figure 4-10 the frequency of each word in a descendent order.

4.3.1 Experiment Setup

All 40 email threads of BC3 have an informative subject and we perform a subject-oriented summarization. The number of words for subject are 5.2 (per email thread), on average.

For extractive summaries, there was not limitation for the number of selected sentences by annotators and they picked many sentences and the number of sentences were not uniform. Nevertheless, we need fix length summaries to be able to compare with other techniques and draw a conclusion. Also, the number of extracted sentences for the whole dataset is 563 (on average for the three annotators) and it is not acceptable to extract more than half of sentences of a dataset as the summary of the dataset.

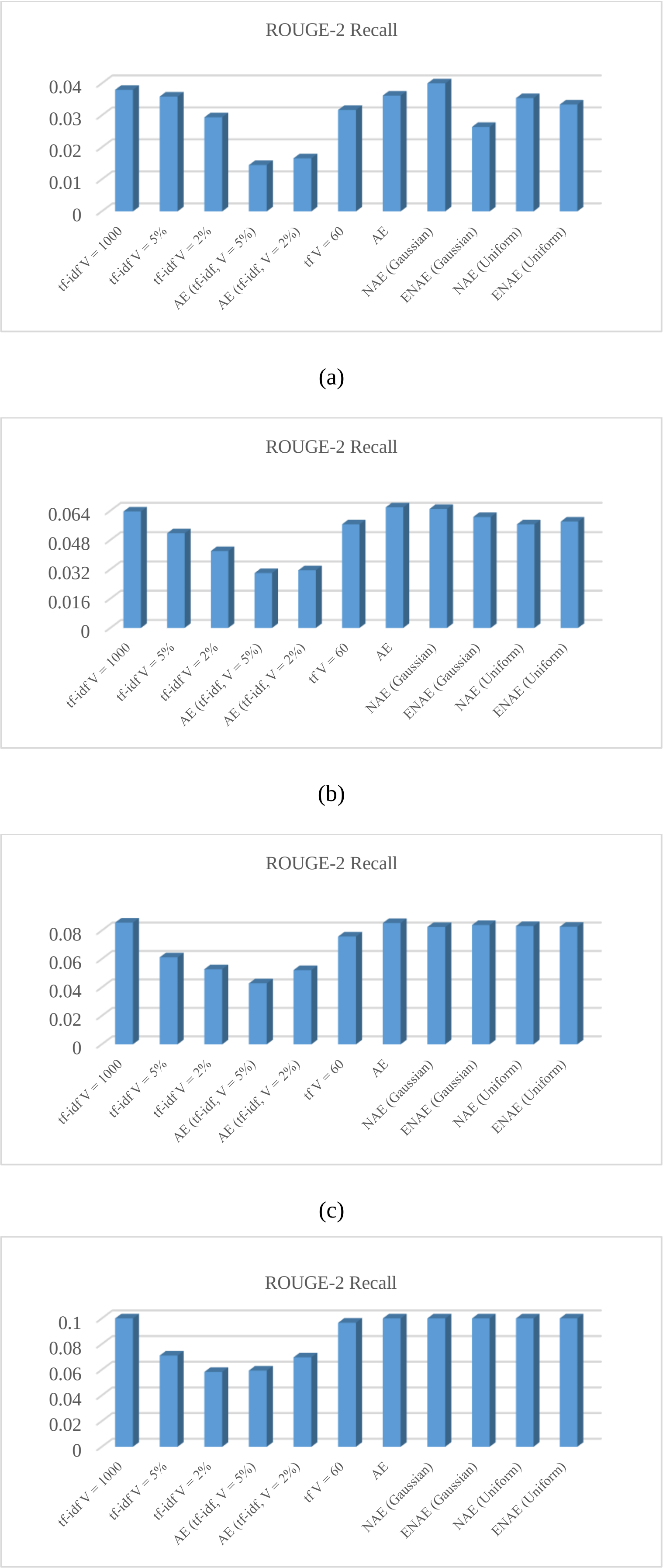
Because the minimum length of the extractive summaries by annotators was 4 sentences, we interpret this length as an agreement of all three annotators that all email threads need at least 4 sentences in their summaries. As a rule of thumb, on average, a summary with 4 sentences will be about 22% of an email thread. Our solution was extracting 4 sentences for each summary and comparing with the abstractive summary as a golden standard generated by human annotator.

BC3 has 2306 unique words and we constructed *tf-idf* vectors based on the 1000, 120 (about 5% of the whole vocabulary) and 60 (about 2% of the whole vocabulary) of the most frequently occurring words. This corresponds to words with a minimum frequency of at least 1, 10 and 16 for vocabulary size 1000, 120 and 60 respectively. For *tf* representation the size of vocabulary is 60.

4.3.2 Results

Figure 4-11 shows how much extracted summaries are close to human generated one. The figure provides the ROUGE-2 recall for summaries with length 1, 2, 3, 4 sentences. Although 4 sentence summaries are the output of the experiment, analyzing summaries with different length may help us to better exploration of the techniques. As can be seen, *tf-idf*, AE (*tf*) and NAE (Gaussian) have fairly good scores compared with other techniques for summaries with only 1 sentence. A key issue is the low score of AE (*tf-idf*), in particular for summaries with less sentences.

This is hardly surprising that the ROUGE score is fairly low for all techniques due comparison of extractive summaries with abstractive ones. Also, the results of AE (*tf-idf*) with smaller vocabulary ($v = 2\%$) is better than larger one ($v = 5\%$). This phenomenon is the same for subject-oriented summarization on SKE.



(a)

(b)

(c)

(d)

Figure 4-11 ROUGE-2 recall score for *tf-idf*, *tf* and AE with different input representations. (a), (b), (c) and (d) are ROUGE scores when the summaries contains 1, 2, 3 and 4 sentences respectively.

Table 4.5 presents exact value of the ROUGE score of *tf-idf* baselines, *tf* and AEs. In summaries with 4 sentences, AE with different representations provides better results compared with *tf-idf* and *tf*.

Although annotators provide abstractive summaries, the model can extract important information. This issue can be concluded from figure 4-12 where only 4 sentences were extracted by the model out of 24 sentences of the email thread. The information such as who is the host, duration (5-2), place and time of meeting (1-2), some people may not attend (6-3), lunch planning (6-3) and another meeting at the same time somewhere else (2-1) are included.

System summary (ENAE):

5-2 I am pleased to announce that Allaire Corp will be our hosts, and look forward to a productive two days, and some time to say hi. The nitty-gritty.
1-2 At the moment we are talking to a potential host in the Boston area about a meeting on thursday and friday 7-8 October.
6-3 Given that there are a small number of us, I thought I would ask what people want to eat for lunch.
2-1 Folks, Please note that this meeting would be on at the same time as the ATIA meeting in Orlando, Florida.

First annotator:

1-1,2 Charles sends an email letting the group know that a face-to-face meeting is necessary and provides a time in October in Boston as a possibility.
3-2 Ian indicates that he can likely attend.
4-1 Kynn replies and states that he will likely be out of the country at the proposed time.
5-4,5,6 Charles then sends some administrative information, including the registration site and tips on booking accommodation in Boston.
6-3,4 Charles also asks the participants for their lunch request.

Second annotator:

1-1 Charles informs the WAI Guidelines committee that there is a meeting planned to resolve issues and revamp the document.
1-2 He mentions a possible meeting in Boston during Oct 7 and 8.
2-1 Charles then adds that the meeting is planned for the same time as the ATIA meeting in Orlando.
2-2 He asks people to let him know if they cannot make the Boston meeting so it can be rescheduled.
3-1 Ian replies that he has other meetings scheduled in Boston at that time.
3-2 He can however probably start attending on Oct 7.
4-1 Kynn says he will be in Rome during that time and thus cannot attend.
5-1 Charles says that the meeting is confirmed for Boston Oct 7 and 8.
5-3,4 Registration is now open at the provided link.
5-5 Hotel rooms are available to be booked.
5-7 He asks people to contact him if they have questions.
6-4,5 Charles mentions a list of possible food choices.
6-3,6,7 He asks people to email him with their choices and if they wish to have certain foods avoided.

Third annotator:

1-1 This thread is compiled from a list of communications from people who would like to have meeting to tidy up their issues.
1-2 The proposal is for the meeting to be in the Boston area on Thursday and Friday of October 7 and 8.
2-1 The meeting is at the same time as the ATIA meeting in Orlando, Florida.
2-2 The schedule may be changed to fit other people's schedules.
3-2 Ian replies saying that he may be able to attend
5-2 Allaire Corp will be the hosts for the meeting.
5-3 meeting registration is open until the first of October through the website, http://www.w3.org/WAI/AU/f2f-oct99
6-3,9 Charles is trying to organize the menu for the meeting.

Figure 4-12 the extracted summary based on subject of the email using ENAE (Gaussian) and three different abstractive summaries by annotators from the sample email thread presented in figure 4-9. The order of sentences is not according to their rankings, but they only show the generated sentences are related to which sentence in the email thread. However, the order of system's result shows the ranking.

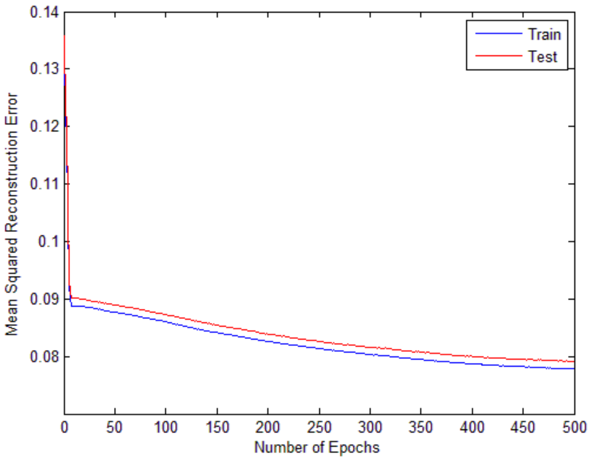
		n = 1	**n = 2**	**n = 3**	**n = 4**
tf-idf V = 1000	(**Recall**)	0.0380	0.0636	0.0856	0.1002
	(Precision)	0.2269	0.1697	0.1582	0.1304
tf-idf V = 5%	(**Recall**)	0.0359	0.0517	0.0612	0.0711
	(Precision)	0.1950	0.1269	0.1032	0.0838
tf-idf V = 2%	(**Recall**)	0.0294	0.0420	0.0528	0.0583
	(Precision)	0.1716	0.1191	0.0942	0.0749
AE (*tf-idf*, *V* = 5%)	(**Recall**)	0.0145	0.0300	0.0429	0.0594
	(Precision)	0.1255	0.1357	0.1157	0.1134
AE (*tf-idf*, *V* = 2%)	(**Recall**)	0.0166	0.0315	0.0522	0.0697
	(Precision)	0.1268	0.1284	0.1278	0.1277
tf V = 60	(**Recall**)	0.0317	0.0566	0.0758	0.0967
	(Precision)	0.1740	0.1321	0.1235	0.1171
AE (*tf*)	(**Recall**)	0.0362	0.0659	0.0853	0.1084
	(Precision)	0.1604	0.1321	0.1239	0.1194
NAE (Gaussian)	(**Recall**)	0.0406	0.0650	0.0825	0.1001
	(Precision)	0.1958	0.1627	0.1442	0.1328
ENAE (Gaussian)	(**Recall**)	0.0264	0.0606	0.0839	0.1017
	(Precision)	0.1491	0.1498	0.1335	0.1288
NAE (Uniform)	(**Recall**)	0.0354	0.0566	0.0832	0.1010
	(Precision)	0.1857	0.1375	0.1319	0.1230
ENAE (Uniform)	(**Recall**)	0.0334	0.0581	0.0827	0.1028
	(Precision)	0.1511	0.1412	0.1304	0.1289

Table 4-5 ROUGE-2 recall of *tf-idf*, *tf* and models versus the number of selected sentences. n is the number of sentences for a summary.

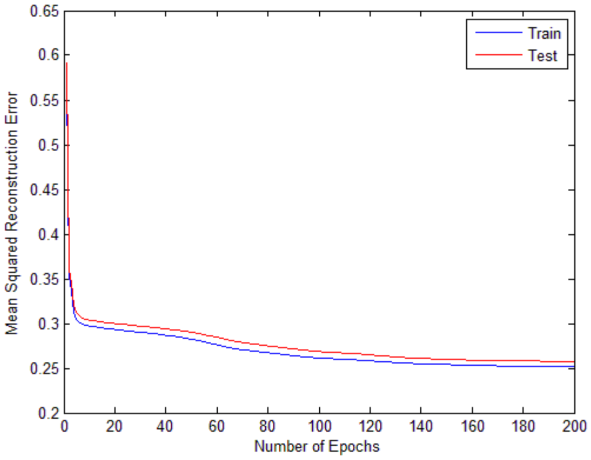
4.4 Generalization

To have a generalized model and avoid over-fitting, it is important to analyze the test set error over the training process. Figure 4-13 shows how AE with *tf* representation as the input can learn the train set and test set. As 10-fold cross-validation approach has been used, the average of these validation errors is presented in figure 4-13 both for train and test set.

As can be seen, the training process of AE with both types of input can optimize the average error. This optimization stopped after 500 iterations for AE and 200 iteration for NAE (Uniform). After training the error of test set is close to train set. With a rule of thumb, there was about 31,430 parameters for training.



(a)



(b)

Figure 4-13 the average squared reconstruction error of the train and test set versus the number of epochs (iterations) for AE (a) and NAE (Uniform) (b) for the SKE corpus.

4.5 Data Comparison

The general trend for both datasets is the same. However, because BC3 contains only 40 emails while SKE contains 349 emails and has more variants of structures, it seems more valuable that we analyze SKE.

Although we hoped that the reduced sparsity stemming from the added noise will improve the results even more, the experiments show that this is not the case – AE (*tf*) (that is without noise) performs mostly better than the noisy AE. However, when combining the rankings of the noisy ensemble, the results are consistent small improvement than a single noisy AE and may even improve over the AE (*tf*). Figure 4-14 shows this improvement.

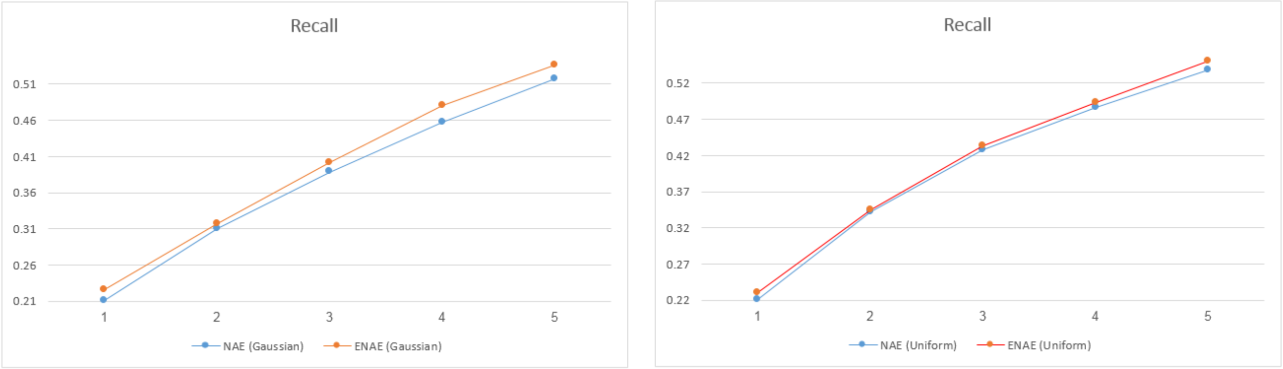


Figure 4-14 ROUGE-2 recall using the keyword phrases as queries. The left panel shows the results based on Gaussian noise. The right panel presents the results based on Uniform noise. The values come from tables 4-3.

Figure 4-15 illustrates the ROUGE-2 recall of the best baseline, *tf* and the AE models with both *tf-idf* and *tf* input representations using keyword phrases as queries and varying the length of the generated summaries versus the extractive summaries of annotators in SKE. While the AE model's results improve almost linearly over the 5 sentences, *tf-idf* gains less from increasing the summary length, particularly from 4 to 5 sentences. The scores are almost the same for *tf*, *tf-idf* and the AE (*tf*) with 1 and 2 sentences. Starting from 3 sentences, the AE performs clearly better compared to *tf-idf*. As can be seen, AE enhance the *tf* representation, specifically when summaries contain more sentences.

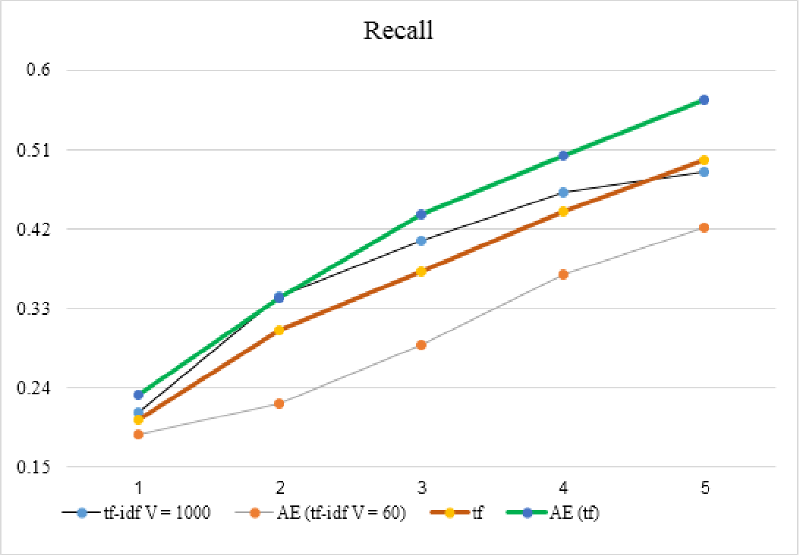


Figure 4-15 ROUGE-2 recall for summaries containing different number of sentences using the keyword phrases as queries. The values come from tables 4-3.

From all experiments of this thesis, the main thing to note is that the AE performs in most cases much better than *tf-idf* baseline, especially when using subjects as queries. The only scenario where *tf-idf* can compete with the AE is with the vocabulary of size 1000 and when using keyword phrases as queries. This is because the manually annotated keyword phrases do in many cases contain the same words as the extracted summary sentences, especially because the queries and summaries where annotated by the same annotators. However, when the vocabulary size is the same as used in the AE, *tf-idf* scores are much lower.

The second thing to notice is that although AE (*tf-idf*) mostly performs better than *tf-idf*, it is still quite a bit worse than AEs derived from the local vocabularies. We believe it is so because the deep AE can extract additional information from the *tf-idf* representations, but the AE learning is more effective when using less sparse inputs, provided by the *tf* representations constructed from local vocabularies. Also, over the all experiments AE (*tf*) performs better than *tf*. This demonstrates that term space maps into a new space that provides more discriminative features.

Chapter 5

5. Conclusion and Future Work

In this thesis, we presented single-document summarization using an unsupervised deep neural network. Our system is query-based, intended to summarize a document, satisfying a request for information expressed by a user's query. We used the deep auto-encoder.

In general, most machine learning techniques for natural language processing (NLP) are limited to numerical optimization of weights for human designed features from the text; our goal however is to learn features.

To evaluate the model, a series of experiments have been conducted on two different publicly available email datasets developed from three different email sources. Experiments on SKE with a wide variety of structures could provide a good explanation on the model. Also evaluations on BC3 as a well-established dataset, but only contains 40 email thread, provided more explorations. To analyze the structure, we performed different threads of experiments: extracting summaries based on the subject of each email, extracting summaries using the annotated keyword phrases as queries and comparing the extracted summaries with extractive and abstractive summaries generated by annotators.

We compared the auto-encoder-based models with the *tf-idf* baseline and showed that AE can enhance *tf-idf* representation. In the experiments, AE (*tf*) performs in most cases much better than the *tf-idf* baseline. The only scenario where the *tf-idf* can compete with the AE is with the vocabulary of size 1000. There is considerable difference between the results when using the email subjects or keyword phrases as queries with keyword phrases leading to better summaries. This was to be expected because the keyword phrases have been carefully extracted by the annotators. The keyword phrases give the highest positive contribution to the *td-idf* baselines with largest vocabularies, which clearly benefits from the fact that the annotated sentences contain the extracted keyword phrases.

Also, using local vocabularies to construct *tf* representation as the input of the AE has been analyzed. Comparing the results of *tf* representation and AE with *tf* representation clearly shows the AE can provide a more discriminative feature space. More precisely, Recall of *tf* representation improves about 11.2%. Indeed, because the goal is to summarize each single document and only the important factor is the semantic similarity of the words of a given document, the AE is capable of mapping this term space into a concept space.

Although we hoped that the reduced sparsity stemming from the added noise will improve the results even more, the experiments show that this is not the case. However, when combining the rankings of the noisy ensemble, the results are much better than a single noisy AE and even in some cases improve over the AE (*tf*). Although after training, an AE applies a deterministic non-linear transformation to compute a new feature space for the data, this ensemble noisy auto-encoder changes the AE from a deterministic feed-forward network to a stochastic model.

In future:

We intend to extend the model for generic summarization. The model is not designed to extract generic information, but with a combination of the model and other techniques this aim can be achieved.

Another idea is to use semi-supervised learning techniques. This type of learning algorithm may help improve the performance.

For a supervised deep structure, noisy labeling may help reduce the manual annotation process. This idea may also complement semi-supervised learning techniques.

In addition to the bag-of-words representation, word-embedding has been successfully used in different NLP tasks. This word representation may help improve and develop the model.

Bibliography

Anderson, J. A. (1995). An introduction to neural networks, MIT press.

Arisoy, E., et al. (2012). Deep neural network language models. Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT, Association for Computational Linguistics.

Baldi, P. and K. Hornik (1989). "Neural networks and principal component analysis: Learning from examples without local minima." Neural networks **2**(1): 53-58.

Barzilay, R. and K. R. McKeown (2005). "Sentence Fusion for Multidocument News Summarization." Comput. Linguist. **31**(3): 297-328.

Bengio, Y. (2009). "Learning deep architectures for AI." Foundations and trends® in Machine Learning **2**(1): 1-127.

Bengio, Y., et al. (2003). "A neural probabilistic language model." the Journal of machine Learning research **3**: 1137-1155.

Bengio, Y. G., Ian J. ; Courville, Aaron (2015). "Deep Learning." MIT Press, ISBN 0-262-13359-8.

Berger, A. and V. O. Mittal (2000). Query-relevant summarization using FAQs. Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics.

Bhaskar, P. (2013). Multi-Document Summarization using Automatic Key-Phrase Extraction. RANLP.

Bishop, C. M. (1995). Neural networks for pattern recognition, Oxford university press.

Blei, D. M., et al. (2003). "Latent dirichlet allocation." the Journal of machine Learning research **3**: 993-1022.

Breiman, L. (1996). "Bagging predictors." Machine learning **24**(2): 123-140.

Burges, C., et al. (2005). Learning to rank using gradient descent. Proceedings of the 22nd international conference on Machine learning. Bonn, Germany, ACM**:** 89-96.

Cao, Z., et al. (2015). Ranking with Recursive Neural Networks and Its Application to Multi-Document Summarization. Twenty-Ninth AAAI Conference on Artificial Intelligence.

Carbonell, J. and J. Goldstein (1998). "The Use of MMR, Diversity-Based Re-ranking for Reordering Documents and Producing Summaries." ACM SIGIR'98: 24-28.

Carreira-Perpinan, M. A. and G. E. Hinton (2005). On contrastive divergence learning. Proceedings of the tenth international workshop on artificial intelligence and statistics, Citeseer.

Collobert, R. and S. Bengio (2004). Links between perceptrons, MLPs and SVMs. Proceedings of the twenty-first international conference on Machine learning, ACM.

Collobert, R., et al. (2011). "Natural language processing (almost) from scratch." the Journal of machine Learning research **12**: 2493-2537.

Conroy, J. M. and D. P. O'leary (2001). Text summarization via hidden markov models. Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, ACM.

Cortes, C. and V. Vapnik (1995). "Support-vector networks." Machine learning **20**(3): 273-297.

Dang, H. T. and K. Owczarzak (2008). Overview of the TAC 2008 update summarization task. Proceedings of text analysis conference.

Das, D. and A. F. Martins (2007). "A survey on automatic text summarization." Literature Survey for the Language and Statistics II course at CMU **4**: 192-195.

Deng, L. and D. Yu (2014). "Deep learning: methods and applications." Foundations and Trends in Signal Processing **7**(3–4): 197-387.

Denil, M., et al. (2014). "Extraction of Salient Sentences from Labelled Documents." arXiv preprint arXiv:1412.6815.

Denil, M., et al. (2014). "Modelling, Visualising and Summarising Documents with a Single Convolutional Neural Network." arXiv preprint arXiv:1406.3830.

Edmundson, H. P. (1969). "New Methods in Automatic Extracting." Journal of The ACM - JACM **16**(2): 264-285.

Erhan, D., et al. (2010). "Why does unsupervised pre-training help deep learning?" the Journal of machine Learning research **11**: 625-660.

Erhan, D., et al. (2009). The difficulty of training deep architectures and the effect of unsupervised pre-training. International Conference on artificial intelligence and statistics.

Erkan, n. and D. R. Radev (2004). "LexRank: graph-based lexical centrality as salience in text summarization." J. Artif. Int. Res. **22**(1): 457-479.

Fawagreh, K., et al. (2015). "On Extreme Pruning of Random Forest Ensembles for Real-time Predictive Applications." arXiv preprint arXiv:1503.04996.

Fletcher, R. and C. M. Reeves (1964). "Function minimization by conjugate gradients." The computer journal **7**(2): 149-154.

Genest, P.-E., et al. (2011). Deep learning for automatic summary scoring. Proceedings of the Workshop on Automatic Text Summarization.

Gong, Y. and X. Liu (2001). Generic text summarization using relevance measure and latent semantic analysis. Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, ACM.

Hagan, M. T., et al. (1996). Neural network design, Pws Pub. Boston.

Hansen, L. K. and P. Salamon (1990). "Neural network ensembles." IEEE Transactions on Pattern Analysis & Machine Intelligence(10): 993-1001.

Hennig, L. (2009). "Topic-based Multi-Document Summarization with Probabilistic Latent Semantic Analysis." Proceedings of the International Conference RANLP: 144-149.

Hinton, G. (2010). "A practical guide to training restricted Boltzmann machines." Momentum **9**(1): 926.

Hinton, G. and R. Salakhutdinov (2011). "Discovering binary codes for documents by learning deep generative models." Topics in Cognitive Science **3**(1): 74-91.

Hinton, G. E. (2002). "Training products of experts by minimizing contrastive divergence." Neural computation **14**(8): 1771-1800.

Hinton, G. E., et al. (2006b). "A fast learning algorithm for deep belief nets." Neural computation **18**(7): 1527-1554.

Hinton, G. E. and R. R. Salakhutdinov (2006a). "Reducing the dimensionality of data with neural networks." Science **313**(5786): 504-507.

Hinton, G. E. and R. S. Zemel (1997). "Minimizing description length in an unsupervised neural network." Preprint.

Hopfield, J. J. (1982). "Neural networks and physical systems with emergent collective computational abilities." Proceedings of the national academy of sciences **79**(8): 2554-2558.

Hovy, E. H. (2005). "Automated Text Summarization." The Oxford Handbook of Computational Linguistics, Oxford: Oxford University Press: 583-598.

Huang, G.-B., et al. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on, IEEE.

Huang, P.-S., et al. (2013). Learning deep structured semantic models for web search using clickthrough data. Proceedings of the 22nd ACM international conference on Conference on information & knowledge management, ACM.

Kågebäck, M., et al. (2014). Extractive summarization using continuous vector space models. Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL.

Kaikhah, K. (2004). "Text summarization using neural networks."

Krysta Svore, L. V., Christopher Burges (2007). "Enhancing Single-document Summarization by Combining RankNet and Third-party Sources." Empirical Methods in Natural Language Processing (EMNLP): 448-457.

Kulkarni, U. and R. S. Prasad (2010). "Implementation and evaluation of evolutionary connectionist approaches to automated text summarization." Journal of Computer Science **6**(11): 1366.

Le, Q. V. and T. Mikolov (2014). "Distributed representations of sentences and documents." arXiv preprint arXiv:1405.4053.

LeCun, Y., et al. (2015). "Deep learning." Nature **521**(7553): 436–444.

Li, J. and S. Li (2014). "A Novel Feature-based Bayesian Model for Query Focused Multi-document Summarization." Transactions of the Association for Computational Linguistics **1**: 89-98.

Li, L., et al. (2009). Enhancing diversity, coverage and balance for summarization through structure learning. Proceedings of the 18th international conference on World wide web, ACM.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. Text Summarization Branches Out: Proceedings of the ACL-04 Workshop.

Liu, Y., et al. (2012). Query-Oriented Multi-Document Summarization via Unsupervised Deep Learning. AAAI.

Louis, A. and A. Nenkova (2013). "Automatically assessing machine summary content without a gold standard." Computational Linguistics **39**(2): 267-300.

Loza, V., et al. (2014). "Building a Dataset for Summarization and Keyword Extraction from Emails." Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14).

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

Mani, I. (2001). Automatic summarization, John Benjamins Publishing.

Manning, C. D. and H. Schütze (1999). Foundations of statistical natural language processing, MIT press.

Maybury, I. M. a. M. T. (1999). "Advances in Automatic Text Summarization." MIT Press, ISBN 0-262-13359-8: 442.

McDonald, R. (2007). A study of global inference algorithms in multi-document summarization. Proceedings of the 29th European conference on IR research. Rome, Italy, Springer-Verlag**:** 557-564.

McDonnell, M. D. and T. Vladusich (2015). "Enhanced image classification with a fast-learning shallow convolutional neural network." arXiv preprint arXiv:1503.04596.

Mikolov, T., et al. (2013). Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems.

Molla, D. and M. E. Santiago-Martinez (2011). Development of a corpus for evidence based medicine summarisation. Proceedings of the Australasian Language Technology Association Workshop.

Nenkova, A. (2005). "Automatic text summarization of newswire: Lessons learned from the document understanding conference." In Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005).

Nenkova, A., et al. (2007). "The pyramid method: Incorporating human content selection variation in summarization evaluation." ACM Transactions on Speech and Language Processing (TSLP) **4**(2): 4.

Ngiam, J., et al. (2011). On optimization methods for deep learning. Proceedings of the 28th International Conference on Machine Learning (ICML-11).

Ouyang, Y., et al. (2011). "Applying regression models to query-focused multi-document summarization." Inf. Process. Manage. **47**(2): 227-237.

PadmaPriya, G. and K. Duraiswamy (2013). "An Approach For Text Summarization Using Deep Learning Algorithm." Journal of Computer Science **10**(1): 1-9.

Palangi, H., et al. "Deep Sentence Embedding Using Long Short-Term Memory Networks."

Porter, M. F. (1980). "An algorithm for suffix stripping." Program **14**(3): 130-137.

Prasad, R. S. K., U. V. Prasad, Jayashree R (2009). "Connectionist Approach to Generic Text Summarization." World Academy of Science, Engineering & Technology(31).

Qazvinian, V., et al. (2010). Citation summarization through keyphrase extraction. Proceedings of the 23rd International Conference on Computational Linguistics, Association for Computational Linguistics.

Radev, D. R., et al. (2000). Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization - Volume 4. Seattle, Washington, Association for Computational Linguistics**:** 21-30.

Rifai, S., et al. (2011). Contractive auto-encoders: Explicit invariance during feature extraction. Proceedings of the 28th International Conference on Machine Learning (ICML-11).

Salakhutdinov, R. and G. Hinton (2009). "Semantic hashing." International Journal of Approximate Reasoning **50**(7): 969-978.

Schmidhuber, J. (2015). "Deep learning in neural networks: An overview." Neural networks **61**: 85-117.

Shewchuk, J. R. (1994). An introduction to the conjugate gradient method without the agonizing pain, Carnegie-Mellon University. Department of Computer Science.

Srivastava, N., et al. (2014). "Dropout: A simple way to prevent neural networks from overfitting." the Journal of machine Learning research **15**(1): 1929-1958.

Tomáš, M. (2012). Statistical Language Models based on Neural Networks, PhD thesis, Brno University of Technology. 2012.[PDF].

Turian, J., et al. (2010). Word representations: a simple and general method for semi-supervised learning. Proceedings of the 48th annual meeting of the association for computational linguistics, Association for Computational Linguistics.

Ulrich, J., et al. (2008). A publicly available annotated corpus for supervised email summarization. Proc. of AAAI email-2008 workshop, Chicago, USA.

Vincent, P., et al. (2008). Extracting and composing robust features with denoising autoencoders. Proceedings of the 25th international conference on Machine learning, ACM.

Welling, M., et al. (2004). Exponential family harmoniums with an application to information retrieval. Advances in neural information processing systems.

Woods, K., et al. (1996). Combination of multiple classifiers using local accuracy estimates. Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on, IEEE.

Wu, H. C., et al. (2008). "Interpreting tf-idf term weights as making relevance decisions." ACM Transactions on Information Systems (TOIS) **26**(3): 13.

Zajic, D. M., et al. (2008). "Single-document and multi-document summarization techniques for email threads using sentence compression." Information Processing & Management **44**(4): 1600-1610.

Zhao, W. X., et al. (2011). Topical keyphrase extraction from twitter. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, Association for Computational Linguistics.

Zhong, S.-h., et al. (2015). "Query-oriented unsupervised multi-document summarization via deep learning model." Expert Systems with Applications **42**(21): 8146-8155.

Appendix

ROUGE-2 Scores

Although in literature ROUGE-2 recall has been referred, the average precision and f-score of the results may help show more information regarding the outcomes. Figure 1, 2 and 3 shows the results of ROUGE-2 for subject-oriented summarization, key-phrase-oriented summarization, and key-phrase-oriented summarization versus extractive summaries, respectively, on SKE dataset. Also, figure 4 presents the results of ROUGE-2 on BC3 dataset.

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.2312	0.2254	0.2221
tf-idf V = 5%	0.1838	0.1764	0.1749
tf-idf V = 2%	0.1435	0.1363	0.1348
tf-idf V = 60	0.1068	0.1041	0.1011
AE (*tf-idf*, *V* = 2%)	0.3580	0.4134	0.3769
AE (*tf-idf*, *V* = 60)	0.3913	0.4429	0.4091
tf V = 60	0.3349	0.3208	0.3222
AE (*tf*)	0.4948	0.5270	0.5056
NAE (Gaussian)	0.4664	0.5029	0.4792
ENAE (Gaussian)	0.5031	0.5264	0.5102
NAE (Uniform)	0.4428	0.5052	0.4673
ENAE (Uniform)	0.4377	0.4867	0.4555

(a)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.2217	0.2188	0.2143
tf-idf V = 5%	0.1688	0.1641	0.1614
tf-idf V = 2%	0.1291	0.1229	0.1217
tf-idf V = 60	0.0935	0.0907	0.0885
AE (*tf-idf*, *V* = 2%)	0.3227	0.3724	0.3384
AE (*tf-idf*, *V* = 60)	0.3422	0.3853	0.3561
tf V = 60	0.3032	0.2949	0.2930
AE (*tf*)	0.4251	0.4457	0.4289
NAE (Gaussian)	0.4033	0.4300	0.4105
ENAE (Gaussian)	0.4325	0.4498	0.4356
NAE (Uniform)	0.3592	0.4121	0.3779
ENAE (Uniform)	0.3659	0.4035	0.3776

(b)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.2062	0.2098	0.2018
tf-idf V = 5%	0.1509	0.1508	0.1459
tf-idf V = 2%	0.1074	0.1095	0.1048
tf-idf V = 60	0.0823	0.0838	0.0800
AE (*tf-idf*, *V* = 2%)	0.2572	0.2980	0.2691
AE (*tf-idf*, *V* = 60)	0.2871	0.3196	0.2958
tf V = 60	0.2452	0.2498	0.2417
AE (*tf*)	0.3360	0.3565	0.3390
NAE (Gaussian)	0.3289	0.3518	0.3335
ENAE (Gaussian)	0.3510	0.3653	0.3520
NAE (Uniform)	0.2618	0.3106	0.2777
ENAE (Uniform)	0.2711	0.3064	0.2813

(c)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.2002	0.2030	0.1968
tf-idf V = 5%	0.1473	0.1500	0.1452
tf-idf V = 2%	0.1048	0.1089	0.1043
tf-idf V = 60	0.0860	0.0901	0.0852
AE (*tf-idf*, *V* = 2%)	0.1951	0.2265	0.2021
AE (*tf-idf*, *V* = 60)	0.2310	0.2567	0.2360
tf V = 60	0.1682	0.1827	0.1701
AE (*tf*)	0.2272	0.2409	0.2273
NAE (Gaussian)	0.2219	0.2368	0.2229
ENAE (Gaussian)	0.2471	0.2569	0.2460
NAE (Uniform)	0.1583	0.1961	0.1701
ENAE (Uniform)	0.1843	0.2077	0.1899

(d)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.1936	0.1984	0.1939
tf-idf V = 5%	0.1453	0.1482	0.1454
tf-idf V = 2%	0.0985	0.1014	0.0986
tf-idf V = 60	0.0859	0.0886	0.0862
AE (*tf-idf*, *V* = 2%)	0.1277	0.1354	0.1291
AE (*tf-idf*, *V* = 60)	0.1805	0.1876	0.1818
tf V = 60	0.1165	0.1213	0.1159
AE (*tf*)	0.1205	0.1286	0.1218
NAE (Gaussian)	0.1067	0.1119	0.1064
ENAE (Gaussian)	0.1370	0.1385	0.1357
NAE (Uniform)	0.0841	0.0912	0.0851
ENAE (Uniform)	0.1140	0.1177	0.1141

(e)

Figure 1 ROUGE-2 for subject-oriented summarization. Table (a), (b), (c), (d) and (e) are the results for summarization with 5, 4, 3, 2, and 1 sentences respectively.

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.4845	0.4817	0.4772
tf-idf V = 5%	0.4217	0.4120	0.4111
tf-idf V = 2%	0.3166	0.3036	0.3041
tf-idf V = 60	0.2224	0.2119	0.2114
AE (*tf-idf*, *V* = 2%)	0.4795	0.5139	0.4909
AE (*tf-idf*, *V* = 60)	0.4220	0.4762	0.4422
tf V = 60	0.4972	0.5134	0.5002
AE (*tf*)	0.5657	0.5556	0.5564
NAE (Gaussian)	0.5179	0.5359	0.5228
ENAE (Gaussian)	0.5370	0.5406	0.5346
NAE (Uniform)	0.5380	0.5228	0.5257
ENAE (Uniform)	0.5504	0.5406	0.5409

(a)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.4607	0.4613	0.4548
tf-idf V = 5%	0.4118	0.4040	0.4022
tf-idf V = 2%	0.3060	0.2942	0.2945
tf-idf V = 60	0.2103	0.2000	0.1998
AE (*tf-idf*, *V* = 2%)	0.4077	0.4324	0.4133
AE (*tf-idf*, *V* = 60)	0.3678	0.4098	0.3813
tf V = 60	0.4395	0.4533	0.4400
AE (*tf*)	0.5035	0.4785	0.4857
NAE (Gaussian)	0.4581	0.4643	0.4558
ENAE (Gaussian)	0.4809	0.4742	0.4724
NAE (Uniform)	0.4872	0.4685	0.4720
ENAE (Uniform)	0.4938	0.4774	0.4798

(b)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.4059	0.4101	0.4007
tf-idf V = 5%	0.3437	0.3478	0.3391
tf-idf V = 2%	0.2655	0.2643	0.2594
tf-idf V = 60	0.1988	0.1895	0.1889
AE (*tf-idf*, *V* = 2%)	0.3179	0.3388	0.3215
AE (*tf-idf*, *V* = 60)	0.2879	0.3303	0.3004
tf V = 60	0.3708	0.3760	0.3664
AE (*tf*)	0.4365	0.4069	0.4147
NAE (Gaussian)	0.3887	0.3966	0.3863
ENAE (Gaussian)	0.4017	0.3974	0.3937
NAE (Uniform)	0.4277	0.4055	0.4100
ENAE (Uniform)	0.4334	0.4100	0.4150

(c)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.3432	0.3414	0.3354
tf-idf V = 5%	0.2874	0.2929	0.2834
tf-idf V = 2%	0.2176	0.2160	0.2115
tf-idf V = 60	0.1661	0.1616	0.1590
AE (*tf-idf*, *V* = 2%)	0.2251	0.2388	0.2260
AE (*tf-idf*, *V* = 60)	0.2226	0.2412	0.2263
tf V = 60	0.3044	0.3064	0.2992
AE (*tf*)	0.3410	0.3225	0.3248
NAE (Gaussian)	0.3107	0.3105	0.3045
ENAE (Gaussian)	0.3168	0.3115	0.3087
NAE (Uniform)	0.3416	0.3209	0.3245
ENAE (Uniform)	0.3450	0.3298	0.3305

(d)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.2217	0.2252	0.2218
tf-idf V = 5%	0.2105	0.2124	0.2099
tf-idf V = 2%	0.1482	0.1486	0.1470
tf-idf V = 60	0.1215	0.1242	0.1215
AE (*tf-idf*, *V* = 2%)	0.1321	0.1372	0.1326
AE (*tf-idf*, *V* = 60)	0.1291	0.1337	0.1297
tf V = 60	0.2034	0.2060	0.2025
AE (*tf*)	0.2397	0.2339	0.2346
NAE (Gaussian)	0.2200	0.2209	0.2187
ENAE (Gaussian)	0.1870	0.1869	0.1852
NAE (Uniform)	0.2547	0.2514	0.2509
ENAE (Uniform)	0.2179	0.2149	0.2145

Figure 2 ROUGE-2 for key-phrase-oriented summarization. Table (a), (b), (c), (d) and (e) are the results for summarization with 5, 4, 3, 2, and 1 sentences respectively.

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.2120	0.1552	0.1733
tf-idf V = 5%	0.1848	0.1310	0.1480
tf-idf V = 2%	0.1424	0.0966	0.1108
tf-idf V = 60	0.0942	0.0629	0.0724
AE (*tf-idf*, *V* = 2%)	0.2115	0.1694	0.1816
AE (*tf-idf*, *V* = 60)	0.1875	0.1632	0.1685
tf V = 60	0.2137	0.1618	0.1784
AE (*tf*)	0.2319	0.1679	0.1895
NAE (Gaussian)	0.2110	0.1647	0.1795
ENAE (Gaussian)	0.2255	0.1676	0.1872
NAE (Uniform)	0.2210	0.1600	0.1801
ENAE (Uniform)	0.2299	0.1675	0.1882

Figure 3 ROUGE-2 for key-phrase-oriented summarization compared with extractive summaries generated by human.

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.1002	0.1304	0.1025
tf-idf V = 5%	0.0711	0.0838	0.0673
tf-idf V = 2%	0.0583	0.0749	0.0604
AE (*tf-idf*, *V* = 5%)	0.0594	0.1134	0.0722
AE (*tf-idf*, *V* = 2%)	0.0697	0.1277	0.0831
tf V = 60	0.0967	0.1171	0.0970
AE (*tf*)	0.1084	0.1194	0.1054
NAE (Gaussian)	0.1001	0.1328	0.1047
ENAE (Gaussian)	0.1017	0.1288	0.1063
NAE (Uniform)	0.1010	0.1230	0.0993
ENAE (Uniform)	0.1028	0.1289	0.1049

(a)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.0856	0.1582	0.1001
tf-idf V = 5%	0.0612	0.1032	0.0670
tf-idf V = 2%	0.0528	0.0942	0.0624
AE (*tf-idf*, *V* = 5%)	0.0429	0.1157	0.0590
AE (*tf-idf*, *V* = 2%)	0.0522	0.1278	0.0691
tf V = 60	0.0758	0.1235	0.0863
AE (*tf*)	0.0853	0.1239	0.0925
NAE (Gaussian)	0.0825	0.1442	0.0955
ENAE (Gaussian)	0.0839	0.1335	0.0928
NAE (Uniform)	0.0832	0.1319	0.0901
ENAE (Uniform)	0.0827	0.1304	0.0939

(b)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.0636	0.1697	0.0844
tf-idf V = 5%	0.0517	0.1269	0.0640
tf-idf V = 2%	0.0420	0.1191	0.0569
AE (*tf-idf*, *V* = 5%)	0.0300	0.1357	0.0468
AE (*tf-idf*, *V* = 2%)	0.0315	0.1284	0.0476
tf V = 60	0.0566	0.1321	0.0726
AE (*tf*)	0.0659	0.1321	0.0814
NAE (Gaussian)	0.0650	0.1627	0.0827
ENAE (Gaussian)	0.0606	0.1498	0.0781
NAE (Uniform)	0.0566	0.1375	0.0702
ENAE (Uniform)	0.0581	0.1412	0.0774

(c)

	Recall	**Precision**	**f-score**
tf-idf V = 1000	0.0380	0.2269	0.0585
tf-idf V = 5%	0.0359	0.1950	0.0538
tf-idf V = 2%	0.0294	0.1716	0.0441
AE (*tf-idf*, *V* = 5%)	0.0145	0.1255	0.0252
AE (*tf-idf*, *V* = 2%)	0.0166	0.1268	0.0283
tf V = 60	0.0317	0.1740	0.0484
AE (*tf*)	0.0362	0.1604	0.0531
NAE (Gaussian)	0.0406	0.1958	0.0590
ENAE (Gaussian)	0.0264	0.1491	0.0416
NAE (Uniform)	0.0354	0.1857	0.0531
ENAE (Uniform)	0.0334	0.1511	0.0512

(d)

Figure 4 ROUGE-2 for subject-oriented summarization on evaluated BC3 dataset. Table (a), (b), (c) and (d) are the results for summarization with 4, 3, 2, and 1 sentences respectively.