Text Summarization Using Unsupervised Deep Learning

Mahmood Yousefi-Azar and Len Hamey
Department of Computing
Faculty of Science and Engineering
Macquarie University, Sydney, NSW, Australia
Email: mahmood.yousefiazar@students.mq.edu.au; len.hamey@mq.edu.au

May 19, 2016

Abstract

We present methods of extractive query-oriented single-document summarization using a deep auto-encoder (AE) to compute a feature space from the term-frequency (*tf*) input. Our experiments explore both local and global vocabularies. We investigate the effect of adding small random noise to local *tf* as the input representation of AE, and propose an ensemble of such noisy AEs which we call the Ensemble Noisy Auto-Encoder (ENAE). ENAE is a stochastic version of an AE that adds noise to the input text and selects the top sentences from an ensemble of noisy runs. In each individual experiment of the ensemble, a different randomly generated noise is added to the input representation. This architecture changes the application of the AE from a deterministic feed-forward network to a stochastic runtime model. Experiments show that the AE using local vocabularies clearly provide a more discriminative feature space and improves the recall on average 11.2%. The ENAE can make further improvements, particularly in selecting informative sentences. 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. We used ROUGE as a fully automatic metric in text summarization and we presented the average ROUGE-2 recall for all experiments.

I. Introduction

Text summarization is an automatic technique to generate a condensed version of the original documents. Manual summarization requires a considerable number of qualified unbiased experts, considerable time and budget and the application of the automatic techniques is inevitable with the increase of digital data available world-wide. The early technique to address with manual text summarization dates back as early as 1958 (Luhn, 1958) and the proposed techniques were reviewed extensively (Lloret and Palomar, 2012; Nenkova and McKeown, 2012).

Text summarization can be categorized into two distinct classes: abstractive and extractive. In the abstractive summarization, the summarizer has to re-generate either the extracted content or the text; however, in extractive category, the sentences have to be ranked based on the most salient information. In many research studies extractive summarization is equally known as sentence ranking (Edmundson, 1969; Mani and Maybury, 1999). In practice, specific text summarization algorithm is needed for different tasks. In particular, a summarization technique can be designed to work on a single document, or on a multi-document. Similarly, the purpose of summarization can be to produce a generic summary of the document, or to summarize the content that is most relevant to a user query. The focus of this paper is to propose an extractive query-oriented single-document summarization technique.

Deep learning showed strong promise in various areas, specifically in natural language processing (NLP) tasks (Collobert et al., 2011; Srivastava and Salakhutdinov, 2012). The pivot of our model is a deep auto-encoder (AE) (Hinton and Salakhutdinov, 2006a) as an unsupervised model. The AE learns the latent representations for both the query and the sentences in the document.

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

The key factor of our model is the word representation. Typically, automatic text summarization systems use sparse input representations. Sparse representations can cause two problems for the model. First, not observing (enough) data in training

Preprint of manuscript accepted 10 October 2016 Please refer to DOI: 10.10.16/j.eswa.2016.10.017

process. This problem is intensified when the selected vocabulary consists of only a subpart of the total presented words in the training data. Second, too much zero in the input and output of AE.

To address these problems, we propose two techniques to reduce sparsity. First, we develop a local word representation in which each vocabulary is designed to construct the input representations of sentences in that document. Second, we add random noise to word representation vector, affecting both the inputs and outputs of the AE.

In our case, after ranking the sentences of a document based on the cosine similarity, they must be selected to generate the summary. Picking the top ranked sentences up is a straightforward selection strategy that is used for AE networks trained without added noise. However, we propose to use an ensemble approach that aggregates the rankings of different experiments, each the result of adding randomly generated noise. Each application uses different random noise added to the input word representation, producing a possibly different extractive summary. The final summary is obtained by selecting the sentences that occur most frequently in the individual summaries. This ensemble approach can make the summarization process robust against small differences between training methods and analogous with manual summarization where annotator(s) would produce different summaries for a document in each review of the document.

The specific contributions of the paper are as follow: We introduce an unsupervised approach for extractive summarization using AEs. Although AEs have 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 by sentence ranking. We will evaluate how AEs handle a sparse word representation such as *tf-idf* and a less sparse word representation based on a document-specific vocabulary, and also the impact of adding random noise to the local word representation vector. The addition of noise changes 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. We show adding stochastic noise to the input and running an ensemble on the same input can make improvements. We expand the email summarization features beyond a set of features and develop to automatically extracted features of emails.

The next section describes recent related work. Section three is dedicated to our model in terms of topology and training algorithms, in particular, Restricted Boltzmann Machine (RBM) and AEs. The word representation is presented in section four. Section five presents sentence ranking measurement detail. The ENAE scheme is described in the six section. After providing experimental setup in the section seven, the results and discussion on the two different email datasets is presented in the section eight. The conclusion and future work is explained in the section nine. Bibliography is placed at the last section.

II. Related Work

Machine learning techniques have widely used for text summarization (Li et al., 2009; Ouyang et al., 2011; Conroy and O'leary, 2001). They are trainable systems in which models learns how to generalize its parameters to extract salient points. 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). 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.

Kaikhah (2004) successfully introduced a shallow neural network for automatic text summarization. Also, Svore et al. (2007) proposed a neural network-based system, called NetSum. This technique was inspired from (Burges et al., 2005). Shardan and Kulkarni (2010) proposed a combination of the multilayer perceptron (MLP) with fuzzy logic and they reported that this combination improves the result. In addition to feed-forward neural networks, recurrent neural networks (RNNs) have been applied for text summarization (Prasad et al., 2009). In spite of different topology, the approach was generally inspired from (Kaikhah, 2004).

Zhong et al. (2015) used a deep architecture that is similar to an AE. They used the learned representations for filtering out not important words of a document in the early layer and discovering key words in later layer. The concept space is to extract the candidate sentences for the summary. However, in our model both the term and concept space are developed based on a sentence of a document. We use the learned representations directly to represent the semantic similar of sentences and in the ranking function. In supervised models, Denil et al. (2014) proposed a model based on a convolutional neural network (CNN) to extract candidate sentences to be included into the summary. A key contribution of the paper is that CNN is trained to classify sentiment labels that are either positive or negative (i.e. a binary label) and sentence extraction can affect the results of predicted a sentiment. Cao et al. (2015) used a recursive neural network for text summarization using hand-crafted word features as inputs. 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. Not using any hand-crafted word representations and labeling data are significant in our proposed model.

The proposed method of this paper in adding noise to the input can be analogous with the denoising auto-encoders (Vincent et al., 2008). To stop overfitting, denoising auto-encoders use noise in the input of AE. In a denoising AE, in order to prevent

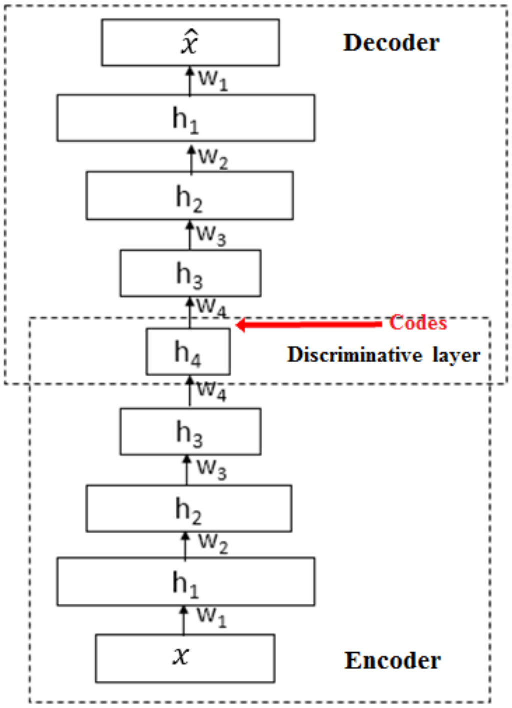


Figure 1: *Our AE topology for dimensionality reduction. The weights of decoder are obtained from unrolled weights of encoder. x and $\hat{x}$ denote the input and reconstructed inputs respectively. h_i are the hidden layers and w_i are the weights. Concept space in which features/codes are extracted $C(x)$ are in this scheme the output of the hidden layer h_4.*

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, first, in most denoising AEs the input is corrupted using a random zero mask while in our model a small noise is added to the input. Second, in our model, the training algorithm adds very small noise to the input but the output is still the same as input. Third, the denoising AE adds noise to the inputs only during training, while we also add noise to input during test time.

III. The Method Description

An AE (Figure 1) is a feed-forward network with the function of learning to reconstruct the input x (Hinton and Zemel, 1997). The AE is indeed trained to encode the input x using a set of recognition weights into a concept space $C(x)$. Then, the features (a.k.a latent representations, codes) from concept space $C(x)$ are converted into an approximate reconstruction of $\hat{x}$ using a set of generative weights. In our model, the generative weights are obtained firstly from unrolled weights of encoder and then from fine-tune phase. Through the encoding process, (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. A deep AE transforms the original input to a more discriminative representation in coding layer (a.k.a discriminative layer or bottleneck) over a hierarchical feature detection in which each level corresponds a different level of abstraction.

Our model has two training stages: pre-training and fine-tune. Pre-training provides an appropriate starting point for the fine-tune phase. That is, obtained weights 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 and Salakhutdinov, 2006a; Hinton et al., 2006b). AEs with pre-training have been successfully applied primarily to documents retrieval (Salakhutdinov and Hinton, 2009; Genest et al., 2011). We apply this technique to shorter texts.

I. Pre-training Stage

Assuming observations are produced from a stochastic generative model in which parameters control the model's behavior (Hinton and Zemel, 1997; Hinton, 2010), Restricted Boltzmann Machine (RBM) as a generative model can be an appropriate

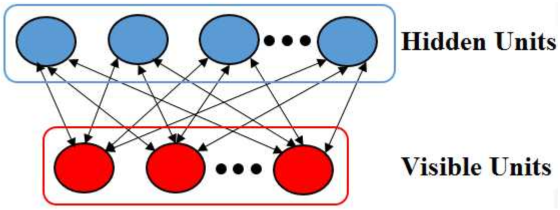


Figure 2: *The schematic representation of the restricted Boltzmann machine (RBM) as a generative model.*

prototype to discover the parameters. The RBM (Figure 2) is the undirected graphical model corresponding a two-layer neural network with one layer of visible units (i.e. input data) and one layer of hidden units (i.e. feature detectors). The symmetric connections of units are restricted between hidden units and visible units, and no visible/hidden unit of the same layer is connected to any other visible/hidden unit. In our layer-wise pre-training, the first RBM is Gaussian-Bernoulli and the other RBMs, except for bottleneck hidden units, are Bernoulli-Bernoulli. In the last RBM, developing to have bottleneck, we replaced binary hidden units with a Noisy Rectified Linear Unit (NReLU) because of presented empirical successfulness in (Nair and Hinton, 2010).

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

$$E(\boldsymbol{x},\boldsymbol{h};\theta) = \quad -\sum_{i\in visible} b_i x_i - \sum_{j\in hidden} a_j h_j \quad -\sum_{i,j} x_i h_j w_{ij} \tag{1}$$

Where w_{ij} is the weights between visible units x_i and hidden units h_j, b_i and a_j are their biases and $\theta = \{\mathbf{W}, \mathbf{a}, \mathbf{b}\}$ represents the network parameters. In terms of the energy function, the joint distribution $p(\boldsymbol{x},\boldsymbol{h};\theta)$ has the following form:

$$p(\boldsymbol{x},\boldsymbol{h};\theta) = \frac{\exp\left(-E(\boldsymbol{x},\boldsymbol{h};\theta)\right)}{Z} \tag{2}$$

Where $Z = \sum_{\boldsymbol{x},\boldsymbol{h}} \exp\left(-E(\boldsymbol{x},\boldsymbol{h};\theta)\right)$ is a partition function as a normalization constant. The marginal probability assigned to a visible vector is the sum over all hidden units:

$$p(\boldsymbol{x};\theta) = \frac{\sum_{\boldsymbol{h}} \exp\left(-E(\boldsymbol{x},\boldsymbol{h};\theta)\right)}{Z} \tag{3}$$

The symmetric structure of the network leads the conditional probabilities for Bernoulli-Bernoulli RBM to be:

$$p(h_j = 1|\boldsymbol{x};\theta) = \frac{\exp\left(\sum_i w_{ij} x_i + a_j\right)}{1 + \exp\left(\sum_i w_{ij} x_i + a_j\right)} = f(\sum_i w_{ij} x_i + a_j) \tag{4}$$

$$p(x_i = 1|\boldsymbol{h};\theta) = \frac{\exp\left(\sum_j w_{ij} h_j + b_i\right)}{1 + \exp\left(\sum_j w_{ij} h_j + b_i\right)} = f(\sum_j w_{ij} h_j + b_i) \tag{5}$$

For visible units with real values, the energy function is:

$$E(\boldsymbol{x},\boldsymbol{h};\theta) = \sum_{i\in visible} \frac{(x_i - b_i)^2}{2\sigma_i^2} \quad - \sum_{j\in hidden} a_j h_j - \sum_{i,j} \frac{x_i}{\sigma_i} h_j w_{ij}, \tag{6}$$

Where σ_i is the standard deviation of the ith visible unit. For Gaussian-Bernoulli conditional probabilities becomes:

$$p(h_j = 1|\boldsymbol{x};\theta) = \frac{\exp\left(\sum_i w_{ij} x_i + a_j\right)}{1 + \exp\left(\sum_i w_{ij} x_i + a_j\right)} = f(\sum_i w_{ij} x_i + a_j) \tag{7}$$

$$p(x_i|\boldsymbol{h};\theta) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x - b_i - \sum_j w_{ij} h_j)^2}{2}\right) = \mathcal{N}\left(\sum_j w_{ij} h_j + b_i, 1\right) \tag{8}$$

Maximum likelihood estimation can be applied to estimate the parameters of the model. To do so, taking the derivative of the negative log-probability of the inputs with respect to the parameters is:

$$\frac{\partial - \log p(x)}{\partial \theta_{ij}} = \frac{\partial}{\partial \theta}(-\log \sum_h p(\boldsymbol{x},\boldsymbol{h})) \tag{9}$$

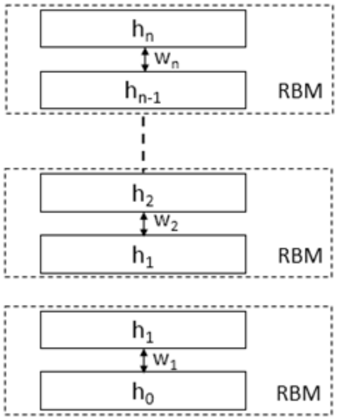


Figure 3: *Multiple RBMs stacked on top of each other.*

$$\frac{\partial - \log p(x)}{\partial \theta_{ij}} = E[\frac{\partial E(\boldsymbol{x}, \boldsymbol{h})}{\partial \theta}|\boldsymbol{x}] - E[\frac{\partial E(\boldsymbol{x}, \boldsymbol{h})}{\partial \theta}] \tag{10}$$

$$\frac{\partial - \log p(x)}{\partial \theta_{ij}} = \langle x_i, h_j \rangle_{data} - \langle x_i, h_j \rangle_{recon} \tag{11}$$

Where the angle brackets are used to denote the expectation of the distribution of the subscript that follows. This leads to a simple learning algorithm by which the parameters update rule is given:

$$\Delta w_{ij} = \epsilon(\langle x_i, h_j \rangle_{data} - \langle x_i, h_j \rangle_{recon}) \tag{12}$$

Where ϵ is a learning rate, $\langle x_i, h_j \rangle_{data}$ is the so-called positive phase contribution and $\langle x_i, h_j \rangle_{recon}$ is the so-called negative phase contribution. The positive phase decreases the energy of observation and negative phase increases the model's energy. However, the computation of the expectation defined by the model is not easily tractable. Hinton (2002) presented 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. In practice, the k-step contrastive divergence (CD-k) approximation provides surprising results (Hinton, 2002).

Carreira-Perpinan and Hinton (2005) numerically compared the update rule with exact log-likelihood gradient with CD-k (Hinton, 2002). Based on this comparison and to have a very good approximation we used CD-1, with running one step Gibbs sampler that is:

$$\boldsymbol{x} = \boldsymbol{x^0} \xrightarrow{p(\boldsymbol{h}|\boldsymbol{x^0})} \boldsymbol{h^0} \xrightarrow{p(\boldsymbol{x}|\boldsymbol{h^0})} \boldsymbol{x^1} \xrightarrow{p(\boldsymbol{h}|\boldsymbol{x^1})} \boldsymbol{h^1} \tag{13}$$

The above mentioned algorithm can be used to have more than one RBMs blocks. To draw a conclusion for the pre-training process, a greedy layer-wise fashion using individual RBMs is employed in which the output of one training RBM is used as an input to the next upper RBM block (Figure 3). In terms of topology, individual RBM blocks would be stacked on top of each other and having the RBM blocks stacked, the topology of the AE with desire deepness meets.

II. Global Adjustment

After extracting the weights of stacked RBMs from the greedy layer-wise unsupervised algorithm (i.e. pre-training), the weights are used as an initialization point of AE with a corresponding configuration. To globally adjust parameters and develop the concept space of our AE, first the stacked RBMs are unrolled (i.e. tied weights). This ensures that the encoder is not operating in the linear regime of its non-linearity (Vincent et al., 2010). Then a fine-tune phase using back-propagation is applied. The fine-tune phase is for the whole network (i.e. all the layers together) and uses a gradient-based optimization algorithm. This approach of using RBM modelling for pre-training followed by back-propagation algorithm for fine-tune is performing two different search methods in the parameter space of the AE. The greedy pre-training learning algorithm performs a search that extracts a good region in the parameter space. In information retrieval applications, it has been observed that deep generative models with only a pre-training phase are comparable with a model that has only one hidden layer (Hinton and Salakhutdinov, 2011). Applying back-propagation from a point in this good region by using a gradient-based local search to fine-tune the model parameter helps take advantage of the multiple hidden layers.

In this unsupervised fine-tuning phase for optimal reconstruction, given the encoding $C(x)$, the whole network tries to minimize the cross-entropy error as the loss function:

$$-\log p(\boldsymbol{x}|c(\boldsymbol{x})) = -\frac{1}{N}\sum_{i=1}^{N} x_i \log \hat{x}_i - \sum_{i=1}^{N}(1-x_i)\log(1-\hat{x}_i) \tag{14}$$

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.

Although stochastic gradient descent (SGD) is a common methodology to fine-tune a neural network, Ngiam et al. (2011) observed that mini-batch Conjugate Gradient (CG) with line search can simplify and speed up training of AEs compared to SGD; thus, we used mini-batch CG with line search and the Polak-Ribiere rule to compute search directions.

IV. Word Representation

A word representation is a type of math object regarding each word. We used a bag-of-words (BOW) representation. This representation does not contain task-specific features and provides a fixed size vector for documents of different sizes. In the BOW representation, a sentence or a document is presented as the collection of its words ignoring the exact word ordering and only retaining the number of occurrences of each word. Intuitively, sentences or documents with similar BOW representations are likely to be similar in content. Although BOW does not embed any semantic information about the words, a neural network with this representation as the input can learn a distributed semantic representation in which semantically related words are relatively close to each other in the generated vector space (Turian et al., 2010).

Term frequency - inverse document frequency (*tf-idf*) is the most commonly used BOW representation for information retrieval and text summarization. It represents each word as an element of a vector comprising of its term frequency in the document, as well as over all documents (*idf*) (Wu et al., 2008). In our model, the *tf-idf* representation is constructed for each sentence as a basis for sentence ranking. Therefore, the input vectors are very sparse since each sentence only contains a small number of words. Sparse representations typically can cause two problems for the leanring algorithm. First, some words may only appear in the test set and never be seen in training. Second, extensive zeros in the output of the AE has an effect on the derivative; i.e. small errors on the zeroes can dominate larger errors on the non-zero outputs when back-propagation modifies the hidden nodes. Genest et al. (2011) and Glorot et al. (2011) presented a re-sampling algorithm to solve this problem for AE.

However, to produce a less sparse representation, we propose using a local vocabulary to construct a locally normalised term-frequency (*tf*) representation. In this representation, the vocabulary for each document is separately constructed from the most frequent terms occurring in that document. The vocabulary size is the same for all documents. This local representation logically has to be less sparse compared to *tf-idf* in which the vocabulary is built from all the documents because the input dimensions correspond only to terms occurring in the current document.

A side effect is that the AE input dimensions now correspond to different words in different documents and the encoder maps the queries and sentences of different documents into different semantic subspaces. However, this causes no adverse effects on the task because our model ranks sentences in each document against that document's individual query.

Also, we added small random values (Gaussian or Uniform distribution) to the *tf* representation to eliminate zeroes 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 absolute zeros any more.

V. Sentence Ranking

Extractive text summarization is also known as sentence ranking. Our model ranks the sentences based on the most relevant and salient characteristics of the documents that are semantically similar to either the subject of an email or previously selected key phrases of a text; thus, this ranking process is an example of query-oriented summarization. The email subject or key phrase is the query text.

Having the AE globally adjusted, the output of the encoder provides the concept space where the codes capture the concepts of a sentence. Because the term space was based on the content of a document and its query, the encoding function maps each sentence into an abstract concept. In the concept space, we use cosine similarity as a common metric. More clearly, the AE learns a low-dimensional concept space. In this space, $c_Q = C(x|Q)$, $c_s = C(x|S_1, \ldots, s_n)$ 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_S is calculated by cosine similarity as following:

$$R(Q, D) = \cos(c_Q, c_S) = \frac{c_Q.c_S}{||c_Q||\;||c_S||} \tag{15}$$

Sentences can be ranked according to the relevance score.

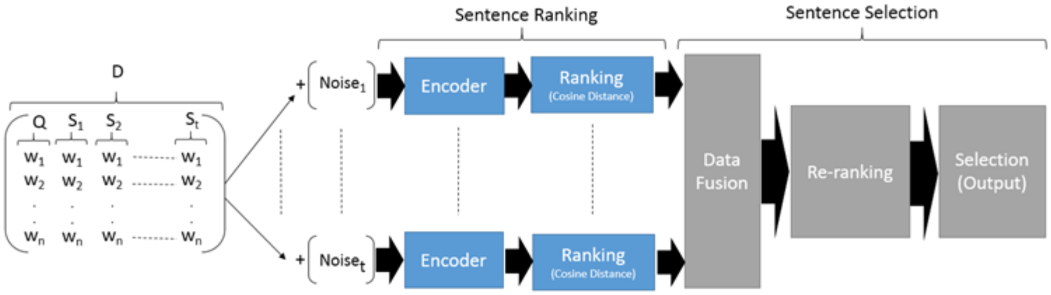


Figure 4: *The Ensemble Noisy Auto-Encoder for query-oriented summarization.*

VI. Iterative Data Fusion and Voting

Ensemble methods use multiple models, mostly classifiers, that are combined to solve a particular problem. In particular, in classifier fusion algorithms, all classifiers are trained in parallel over the entire feature space (Hansen and Salamon, 1990). However, in particular, Bucilua et al. (2006) have presented a neural network model by which a large and complex ensemble can be compressed into a smaller and faster model. Hinton et al. (2015) developed this model compressing method further using a different compressing technique and showed that an ensemble of deep neural networks can be distilled into a single deep network while the quality of trained ensembles is retained.

Although our model is a single deep network, the proposed technique is not distilling an ensemble of models into a single model but developing a single model from scratch using an ensemble of runs on the same network. Figure 4 presents the proposed model. It is an integrated system in which semantically similar sentences to a query can be ranked. 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.

In detail, after ranking the sentences of a document based on the cosine distance to the given query, a number of them must be selected to be included into the summary. A straightforward selection strategy is just to select the top ranked sentences. We use this selection strategy for the AE without additive noise to the input. However, with the noisy input representations introduced in the word representation section, an experiment can be repeated using the same input vector but with differently generated noise several times. This iterative ranking (i.e. each experiment) provides a variations in ranking.

Then, an ensemble approach that aggregates the rankings of different experiments on the same input document is used to fuse the ranking. Specifically, we use a majority voting scheme to fuse the ranks. Since the raw rankings have been obtained using additive noise, the voting algorithm ignores the rank and only considers the number of times that each sentence appears in the top group. 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: $S_{k-1} = \{\hat{s}_1, ..., \hat{s}_{k-1}\}$. Where $S_k = S_{k-1} \cup \{\hat{s}_k\}$ with the following formula: $\hat{s}_k = \arg\max_{s_k \in D | S_{k-1}} [score(s_k)]$ (where $score(s_k)$ is the number of times s_k is selected by each component).

VII. Experimental Setup

To have a extensive exploration of the model, we conducted different experiments on the two different publicly available email datasets that are specifically developed for summarization: Summarization and Keyword Extraction from Emails (SKE) (Loza et al., 2014), BC3 from British Columbia University (Ulrich et al., 2008).

SKE consists of 349 emails, of which 319 are from the Enron mailboxes and 30 were provided by volunteers. The email sets contain both private and corporate emails and are divided into two parts: single emails of at least 10 sentences and threads containing at least 3 emails. In total, SKE consists of more than 100,000 words. Stemming using Porter (1980) and stop-word removal [1] reduces this to 46,603 words comprising 7478 unique words. There are 6801 sentences — an average of 19.5 sentences per email. The average email is 303 words and the average sentence is 15.5 words.

Two different annotators provided an abstractive summary, an extractive summary and a set of key phrases for each email. For extractive summaries, the annotators identified 5 different sentences in a ranking fashion; therefore, in our experiments with this dataset, generated summaries have 5 sentences. The abstractive summaries are between 33 and 96 words long, averaging 73 words per email. Key phrases were restricted to 4 words, averaging 2.1 words per phrase.

We conducted two different query-oriented summarization experiments on the SKE corpus. The first experiment used the email subject as the query text — this experiment could only be performed on the 289 emails that have non-empty subjects.

[1] obtained from http://xpo6.com/list-of-english-stop-words

The second experiment used keyword phrases, which are available for the whole dataset, as the queries.

Subject: Nissho NDA

01 Kobayashi-san, Sorry for the delay.
02 I am working through a few issues here and I expect to get back to you shortly.
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.
05 Please review it and it would be great if you could make original again and send us for Mr.Miyazawa's signature.
06 Also, please advise us how we should treat Nissho's NDA in these circumstances.
07 BTW, I talked with Mr.Matsubara, Manager of Enron Global Finance in Tokyo.
08 We are internally discussing when we start our official meeting.
09 Please let us know when you are ready.
10 Please approve or make changes to their new NDA.
11 They need to change the counterparty name due to a joint venture.
12 I wanted to let you know this was coming in as soon as Mark approves the changes.
13 Dealing with Japan is somewhat time challenging.

Subject: Next face to face meeting

01 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).
02 At the moment we are talking to a potential host in the Boston area about a meeting on thursday and friday 7-8 October.
03 More details as they come to hand.
04 Folks, Please note that this meeting would be on at the same time as the ATIA meeting in Orlando, Florida.
05 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).
06 Hello, I have other meetings tentatively scheduled for 6 and 7 October in Boston.
07 I would probably be able to attend starting the 7th.
08 I believe that I will be in Rome on the 6th and 7th.
09 Dear working group participants, As discussed, the meeting will indeed be held on thursday 7 and friday 8 October.
10 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.
11 Meeting registration is open from now until 1 October.
12 register via the meeting page at http://www.w3.org/WAI/AU/f2f-oct99 which also provides details on location, accommodation, etc.
13 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.
14 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.
15 If you have any questions or special requirements please contact me as early as possible.
16 Most important if you haven't registered yet, don't forget.
17 http://www.w3.org/WAI/AU/f2f-oct99 for the details.
18 Given that there are a small number of us, I thought I would ask what people want to eat for lunch.
19 The default choices are roast turkey breast with vegetables (without turkey if you are a vegan), and pasta with vegetable rataouille.
20 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.
21 I will do this in the following way: If you cannot eat one of these things please make that very clear.
22 It will take at least two votes to change something, and will have to be compatible with what people can't do.
23 The deadline is 3pm (boston time) Monday.
24 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 5: *A randomly selected sample email thread from SKE, corporate section (ECT020.xml) and from BC3 (list number: 058-13714734). The subject of SKE email is ”Nissho NDA”. The subject of BC3 email is ”Next face to face meeting”.*

The BC3 data was collected from mailing lists that are mostly not as informal as ordinary emails. The dataset contains 40 email threads each of which has less than 11 emails and less than 6 participants, on average. The dataset contains 3222 sentences including repetitions due to quoting. Removing repetition, signatures and sender's name leaves 736 sentences, an average of 18.4 sentence per email thread. Stemming and stop-word removal further reduce the data set from 15,000 to 7101 words with 2306 unique words. The average number of words per email thread is 380 and per sentence is 21.

BC3 has been annotated by 3 different annotators for both extractive and abstractive summarization. However, the extractive summaries are not restricted to a limited number of sentences and do not involve sentence ranking so they are not suitable as goals for our approach. The shortest extractive summaries were 4 sentences, so our experiments produce 4 sentences as extractive summaries for this dataset. These summaries are compared with the annotators' abstractive summaries using ROUGE-2 as explained below. The abstractive summaries were limited to 250 words and are on average 122 words for each email thread. All 40 email threads of BC3 have an informative subject which we use as the query text for summarization.

The average subject length is 5.2 words.

Our baseline is *tf-idf* (with the inverse document frequency as the *idf* term) and over experiments we use several different vocabulary sizes. *tf-idf* is a term based technique and a case study helps better undestanding of our model regarding meaning extraction. Figure 5 shows randomly selected emails from SKE and BC3 for the case study. Because there is no general agreement on a vocabulary size in the literature we decided to choose vocabulary sizes by which two datasets can be comparable and also providing different test settings to evaluate the proposed model. The smallest vocabulary, 60 words, is selected because Genest et al. (2011) used less than 1.5% of the unique words of a large dataset like Text Analysis Conference (TAC) corpus for the input of an AE and we want to provide at least 2% of the unique words. The largest vocabulary of 1000 words was selected because assuming words with frequency 1 as idiosyncratic words, in BC3, only 1000 unique words have the frequencies of at least 2. In short, we constructed vocabularies based on the 1000, 374/120 (about 5% of the whole vocabulary of SKE/BC3), 150 (about 2% of the whole vocabulary of SKE), and 60 most frequently occurring terms (around 2% of the whole vocabulary of BC3).

The selection of a metric for comparison of the generated summary with the human summary is still an open question (Louis and Nenkova, 2013). We used ROUGE (Lin, 2004) package as a fully automatic metric for evaluating the performance of our model. The pyramid metric is an indirect manual evaluation method (Nenkova et al., 2007) that has previously been used for BC3 (Ulrich et al., 2009). However, due to lack of resources we used the fully automatic metric. Of the different forms of ROUGE, we use ROUGE-2 Recall because Dang and Owczarzak (2008) observed that it has the highest correlation with manual scores. In all our experiments, the confidence interval was 95% and a jackknifing procedure was used for multi-annotation evaluation Lin (2004). We used 10-fold cross-validation with 90% training data.

The AE has 140 units in the first hidden layer, then 40 units, 30 units and 10 units as the bottleneck (i.e. codes). The decoder portion uses the reverse pattern. We used the same architecture for all the different vocabulary sizes. We used G Hinton's training algorithms [2]. The size of the first hidden layer (140 units) was chosen to be at least double the smallest vocabulary size of 60 inputs, because this may help RBM to obtain a better representation (Bengio, 2009; Erhan et al., 2009; Le Roux and Bengio, 2008). We chose to include two additional hidden layers with 40 and 30 units because (Hinton and Salakhutdinov, 2006a) showed in the supplementary material paper [3] that a pre-trained *deep* AE can have better performance than a pre-trained *shallow* AE with the same number of parameters. The ten code units provide a ten-dimensional concept space to represent the semantics of each sentence. Theoretically, learning a desired concept in a large concept space can be as difficult as finding a needle in the haystack and experimental results show having more units may negatively impact on the AE for summarization (Zhong et al., 2015).

We used the same hidden units structure throughout our experiments for two reasons. First, this makes the results more comparable. Second, an AE with more parameters to be tuned would require more manually annotated data that was not available to us. Also, we used the same hyper-parameters, obtained on the basis of presented practical guides in (Bengio, 2012; Hinton, 2010), over the all experiments.

We explored several input representations: *tf-idf*, *tf* using local vocabularies (*Ltf*), *tf-idf* and *Ltf* as the input of AE, *Ltf* with added Gaussian/Uniform noise (Ltf-NAE) and in the noisy ensemble *Ltf* (Ltf-ENAE) is used for the input of the AE. Both additive Gaussian and Uniform noise have the same variance over the all experiments.

VIII. Results and Discussion

I. Subject-oriented Summarization with SKE

The ROUGE-2 recall of the *tf-idf* baselines and proposed models is presented in table 1. The table presents the performance of models in extracted summaries with different length. This is hardly surprising that recall improves with more longer summaries and larger vocabulary size for *tf-idf*. This improvement is relatively constant for our baselines. Using *tf-idf* as the input of the AE, the recall is improved by increasing the number of sentences in the summary. However, increasing the vocabulary size provides less effective AE representation and reduces the recall, possibly because the larger vocabulary increases the proportion of zero inputs.

Comparing *tf-idf*, *Ltf*, AE (*tf-idf*) and AE (*Ltf*) shows three important points. First, AE (*Ltf*) is always much better than *Ltf*. The relative difference is greater for longer summaries, demonstrating that AE enriches the representation and provides a more discriminative space. Second, AE enhances the discriminative power of both *tf-idf* and *Ltf*, but the recall of AE (*Ltf*) is higher. As above, *Ltf* provides less sparse inputs than *tf-idf*, and the AE learns more effectively. Finally we note that AE (*Ltf*) outperforms the *tf-idf* baselines even when *tf-idf* uses much larger vocabulary sizes.

The table shows that in general, the ROUGE score of AE (*Ltf*), Ltf-NAE and Ltf-ENAE, both perturbed with either Gaussian or Uniform noise, is itself noisy (i.e. having no deterministic order) for summaries with only one sentence. However,

[2] http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html

[3] http://www.cs.toronto.edu/~hinton/absps/science_som.pdf

Model	**n = 1**	**n = 2**	**n = 3**	**n = 4**	**n = 5**
tf-idf V=1000	0.1936	0.2002	0.2062	0.2217	0.2312
tf-idf V=5% (374 words)	0.1453	0.1473	0.1509	0.1688	0.1838
tf-idf V=2% (150 words)	0.0985	0.1048	0.1074	0.1291	0.1435
tf-idf V=60	0.0859	0.0860	0.0823	0.0935	0.1068
AE (*tf-idf* V=2%, 150 words)	0.1277	0.1951	0.2572	0.3227	0.3580
AE (*tf-idf* V=60)	0.1805	0.2310	0.2871	0.3422	0.3913
Ltf V=60	0.1165	0.1682	0.2452	0.3032	0.3349
AE (*Ltf* V=60)	0.1205	0.2272	0.3360	0.4251	0.4948
Ltf-NAE (Gaussian)	0.1067	0.2219	0.3289	0.4033	0.4664
Ltf-ENAE (Gaussian)	0.1370	0.2471	0.3510	0.4325	0.5031
Ltf-NAE (Uniform)	0.0841	0.1583	0.2618	0.3592	0.4428
Ltf-ENAE (Uniform)	0.1140	0.1843	0.2711	0.3659	0.4377

Table 1: *ROUGE-2 recall of subject-oriented summarization for* tf-idf*,* tf *and models versus the number of selected sentences.* n *is the number of sentences for a summary.*

with the increase of the number of extracted sentences, the scores of all variants of AEs are relatively closer to each other, in particular for summaries with 5 sentences. Among AE-based techniques, more relevant sentences can be extracted using Ltf-ENAE (Gaussian) and AE (*Ltf*) compared to Ltf-NAE (with either Gaussian or Uniform) and Ltf-ENAE with voted from Uniform noise runs. In extracted summaries with one sentence, the highest recall belongs to *tf-idf* whereas for summaries with more sentences AE-based models preform much better. This can be interpreted that in subject-based summarization mapping term space to concept space is not quite accurate for every single vector, but the mapping is much accurate on average (i.e. the summaries with more sentences).

To further evaluate our model, we analyse the results of Ltf-NAE and Ltf-ENAE generated from the sample email thread (Figure 5). The first annotator selected and ranked sentences 3, 4, 10, 11 and 5 while the second annotator selected and ranked sentences 3, 4, 11, 6 and 5. For comparison, Ltf-NAE and Ltf-ENAE ranked sentences number 4, 10, 3, 6, 12 and 3, 8, 4, 6, 11, respectively, as the top 5 sentences. Both generated summaries contain sentences 03, 04 with high rank similar to annotators. Sentence 11 is selected by both annotators at a middle rank and is selected by Ltf-ENAE but not by Ltf-NAE in this particular example. Sentences 10 and 6 are each selected by only one annotator and are ranked lower by both algorithms. The only annotator selected sentence that is not ranked by the algorithms is sentence 5, which was ranked last by both annotators.

II. Key-phrase-oriented Summarization with SKE

Key phrase extraction has been used for text summarization (Qazvinian et al., 2010; Zhao et al., 2011). Where the annotators have provided key phrases, we can use them as the query. Table 2 shows the ROUGE-2 recall of the baselines and AE-based models with various input representations. Since the summarization uses carefully extracted key phrases, recall increased as expected. Although this improvement is for all techniques, the amount of improvement varies for each technique.

AE (*tf-idf*) provides better performance than plain *tf-idf* with the same size vocabulary. Considering *n=5* which has the highest performance, *tf-idf* with vocabulary sizes 150 and 60 yields recalls of 0.3166 and 0.2224 respectively which are improved by AE (*tf-idf*) to 0.4795 and 0.4220. It is worth noting that the larger vocabulary size here yields better recall, in contrast with the subject-oriented summarization results (Table 1). We hypothesize that the enriched term space of the key phrase queries enables AE, as an unsupervised model, to learn a richer concept space, yielding richer summaries.

The AE (*Ltf*) performance is better than *Ltf*. There is a fairly important difference between subject-oriented and key-phrase-oriented summarization for SKE corpus when *n=1*. The ROUGE score of the AE with various *Ltf*-based input representations is higher than *tf-idf* and AE (*tf-idf*). *Ltf* uses a local vocabulary for each e-mail, meaning that there is a greater likelihood of the words in the key phrases being present in the vocabulary. Words that are not in the vocabulary cannot contribute to the concept space, so *Ltf* supports a richer concept space for small vocabulary sizes.

Table 2 shows that AE (*Ltf*) and ENAE (Uniform) provide better results in comparison with other techniques for values of *n* between 3 and 5. Comparing the performance of ENAEs with the corresponding NAEs, using either subject or key phrase, shows a small improvement in most cases. We hypothesize that the ensemble regularises the concept space, producing a more effective concept mapping irrespective of the information content.

Our case study (Figure 5) also shows that summarization using key phrases with Ltf-NAE and Ltf-ENAE provide better results compared with their subject-oriented counterparts. Ltf-NAE provides a good ranking in which 3 sentences are ranked

Model	**n = 1**	**n = 2**	**n = 3**	**n = 4**	**n = 5**
tf-idf V=1000	0.2217	0.3432	0.4059	0.4607	0.4845
tf-idf V=5% (374 words)	0.2105	0.2874	0.3437	0.4118	0.4217
tf-idf V=2% (150 words)	0.1482	0.2176	0.2655	0.3060	0.3166
tf-idf V=60	0.1215	0.1661	0.1988	0.2103	0.2224
AE (*tf-idf* V=2%, 150 words)	0.1321	0.2251	0.3179	0.4077	0.4795
AE (*tf-idf* V=60)	0.1291	0.2226	0.2879	0.3678	0.4220
Ltf V=60	0.2034	0.3044	0.3708	0.4395	0.4972
AE (*Ltf* V=60)	0.2397	0.3410	0.4365	0.5035	0.5657
Ltf-NAE (Gaussian)	0.2200	0.3107	0.3887	0.4581	0.5179
Ltf-ENAE (Gaussian)	0.1870	0.3168	0.4017	0.4809	0.5370
Ltf-NAE (Uniform)	0.2547	0.3416	0.4277	0.4872	0.5380
Ltf-ENAE (Uniform)	0.2179	0.3450	0.4334	0.4938	0.5504

Table 2: *ROUGE-2 recall of key-phrase-oriented summarization for* tf-idf, tf *and models versus the number of selected sentences.* n *is the number of sentences for a summary.*

in the top by both annotators and the other 2 by one annotator. Ltf-ENAE also has 3 sentences in common with both annotators while one other sentence has also been selected by one of annotators. The only sentence that is unranked by the annotators is sentence number 12 — we will see in the next subsection that this sentence is also informative.

III. Key-phrase-oriented Summarization Versus Abstractive Summary with SKE

SKE also provides annotators' abstractive summaries. These can be used as a 'gold standard' to compare our model's summaries with human abstracts. Table 3 present the ROUGE-2 scores of the extracted sentences versus abstractive summaries. It is to be expected that the recall will decrease to small values because the abstractive summaries are written by humans and only contain the abstract meaning of the emails, not the exact sentences. Despite this, the trend is very similar to the previous subsection. This experiment reinforces that AE can provide a more discriminative concept space and that Ltf-ENAE (Gaussian/Uniform) mostly yield better results compared other baselines.

Model	**Recall**
tf-idf V=1000	0.2120
tf-idf V=5% (374 words)	0.1848
tf-idf V=2% (150 words)	0.1424
tf-idf V=60	0.0942
AE (*tf-idf* V=2%, 150)	0.2115
AE (*tf-idf* V=60)	0.1875
Ltf V=60	0.2137
AE (*Ltf* V=60)	0.2319
Ltf-NAE (Gaussian)	0.2110
Ltf-ENAE (Gaussian)	0.2255
Ltf-NAE (Uniform)	0.2210
Ltf-ENAE (Uniform)	0.2299

Table 3: *ROUGE-2 recall for* tf-idf, tf *and models, abstractive summaries generated by human versus extractive summaries*

First Annotator

- 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.

Second Annotator

- 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.

Figure 6: *the generated abstractive summaries of sample email (Figure 5) by two annotators.*

Figure 6 presents the generated abstractive summaries by annotators. Taking the sentence number 12 into consideration, although the sentence is not among extracted sentences by Ltf-ENAE, it contains important information that both annotators have pointed out.

IV. Subject-oriented Summarization with BC3

As discussed previously, only the abstractive summaries of BC3 are appropriate for our study. Table 4 presents the ROUGE-2 recall of extracted summaries compared with the three annotators' abstractive summaries; thus, it is expected that recall scores are small values. While each extracted summary contains 4 sentences, for completeness we also evaluate summaries of 1, 2 and 3 sentences.

In table 4, AE (*tf-idf*) performs worse than the other approaches, in contrast with the previous results in tables 1, 2 and 3. This may come from the relative difference between SKE and BC3. More clearly, the number of emails in BC3 is about 12% of SKE while the number of unique words in BC3 is about 30% of SKE. This results in a more sparse *tf-idf* representation in BC3 than SKE. Similar to the subject-oriented summarisation of SKE, the performance of AE (*tf-idf*) is better with the smaller vocabulary (*n=2%*). For small vocabularies, the best results are obtained using variants of AE with *Ltf*. As in the above subsections, this performance improvement is related to the local vocabulary providing a richer representation of both the query and text.

For the case study presented in figure 5, we again consider the output of Ltf-NAE and Ltf-ENAE. Although in this randomly selected email, there is no intersection between the top ranked sentences of Ltf-NAE and Ltf-ENAE, both models do extract important information from this long email thread. Ltf-NAE extracted information such as possible menu substitutions (sentence 20), flexibility of schedule changes (sentence 5) and avoiding certain foods (sentence 19); the non-informative sentence 15 was also included in this extract. Ltf-ENAE selected very important information, that is highly correlated with annotator's summaries, such as the hosting organisation and duration (sentence 10), place and time of meeting (sentence 2), some people may not attend and lunch planning (sentence 18) and a competing meeting (sentence 4).

Model	**n = 1**	**n = 2**	**n = 3**	**n = 4**
tf-idf V=1000	0.0380	0.0636	0.0856	0.1002
tf-idf V=5%	0.0359	0.0517	0.0612	0.0711
tf-idf V=2%	0.0294	0.0420	0.0528	0.0583
AE (*tf-idf* V=5%)	0.0145	0.0300	0.0429	0.0594
AE (*tf-idf* V=2%)	0.0166	0.0315	0.0522	0.0697
Ltf V=60	0.0317	0.0566	0.0758	0.0967
AE (*Ltf* V=60)	0.0362	0.0659	0.0853	0.1084
Ltf-NAE (Gaussian)	0.0406	0.0650	0.0825	0.1001
Ltf-ENAE (Gaussian)	0.0264	0.0606	0.0839	0.1017
Ltf-NAE (Uniform)	0.0354	0.0566	0.0832	0.1010
Ltf-ENAE (Uniform)	0.0334	0.0581	0.0827	0.1028

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

V. Generalization and Data Comparison

The generalization capability for deep structures with millions of parameters is quite crucial. Poor generalization arises from over-fitting of the training data, where the error on the training set is significantly small while this error is still large for test set (i.e. new data). To address this problem, AEs employ training algorithms such as a sparsity constraint, contractive mapping, dropout, de-noising, early stopping, L2 regularization (i.e. weight-decay) and unsupervised pre-training (Srivastava et al., 2014; Rifai et al., 2011; Vincent et al., 2008; Erhan et al., 2010). While a gradient-based algorithm gradually increases weights, corresponding to a regularized learning process (Collobert and Bengio, 2004), early stopping alone is not sufficient enough to prevent over-fitting. Our experiments use a combination of early stopping, L2 regularisation and unsupervised pre-training. Figure 7 demonstrates that both the training and test errors are decreasing throughout training, and that the trained AE has good generalisation.

The two reasons for adding small random noise to the input representation were to eliminate zeroes and to support ensembles. The results show that eliminating zeroes alone is not beneficial — the performance of AE (*Ltf*) is typically better than Ltf-NAE. The comparison is based on the SKE data because it is larger and more diverse. Figure 8 illustrates the improvement of summarization provided by AE(*Ltf*), particularly for summaries of 5 sentences. Comparing AE(*Ltf*) with *tf-idf* in this figure, we note that AE(*Ltf*) improves linearly with increasing the number of sentences in a summary, whereas *tf-idf* gains less from larger summaries.

To evaluate ensembles, we compare Ltf-NAE with Ltf-ENAE. Figure 9 shows that the Ltf-ENAE technique usually outperforms Ltf-NAE. Ltf-ENAE scheme may even improve on AE(*Ltf*).

The results particularly confirm the value of AEs. The AE-based models generally perform much better than the *tf-idf* baselines in both the SKE and BC3 data sets — only *tf-idf* with the vocabulary of size of 1000 can compete with the AE models. This demonstrates that the AE models are more capable of encoding the information content of a small vocabulary.

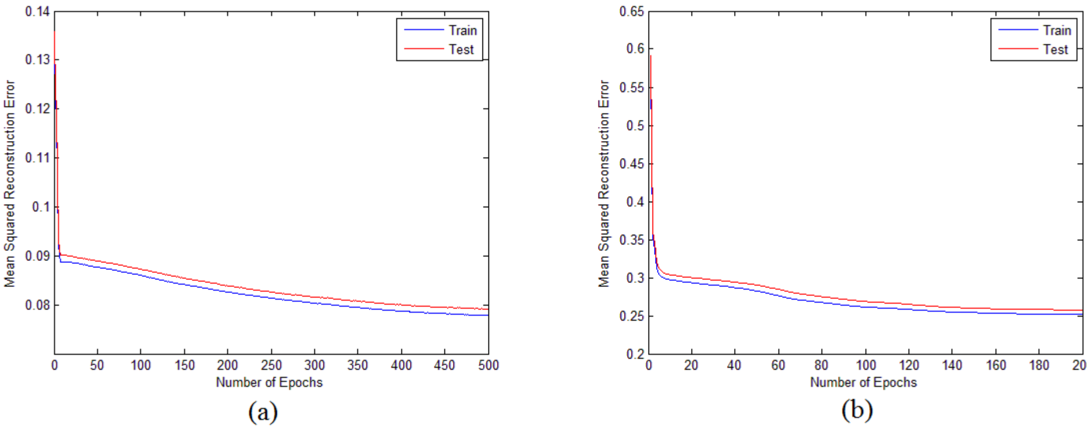


Figure 7: *the average squared reconstruction error of the train and test set versus the number of epochs (iterations) for AE(*Ltf*) (a) and Ltf-NAE (Uniform) (b) for the SKE corpus.*

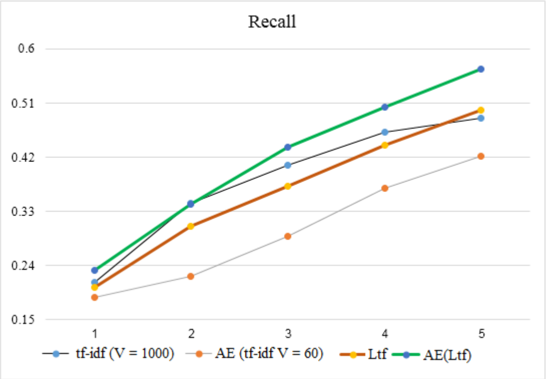
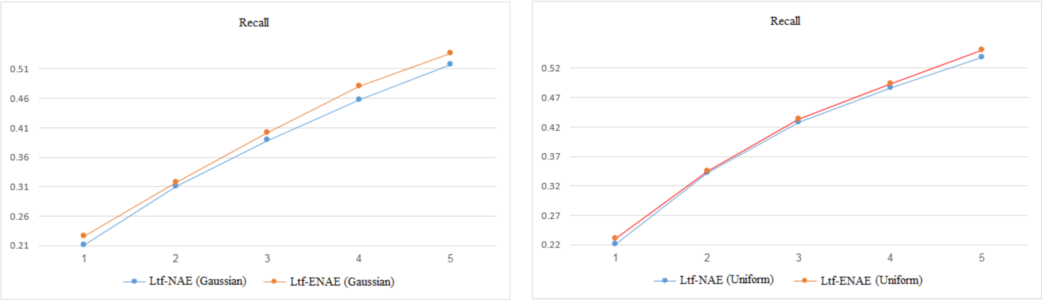


Figure 8: *the ROUGE-2 recall of the best baseline* td-idf, tf, *the AE (*tf-idf*) and AE(*Ltf*) using keyword phrase as query.*



(a) (b)

Figure 9: *ROUGE-2 recall using the keyword phrases as queries. (a) shows the results based on Gaussian noise. (b) presents the results based on Uniform noise.*

IX. Conclusion and Future Work

In this paper, we presented a query-based single-document summarization scheme using an unsupervised deep neural network. We used the deep auto-encoder (AE) to learn features rather than manually engineering them. A series of experiments were conducted on the SKE and BC3 email datasets, using the subject and keyword phrases as queries. We compared using global and local vocabularies to construct word representations as the input of the AE. Comparing the results of local term frequency (*Ltf*) with and without AE demonstrates that the AE provides a more discriminative feature space in which semantically similar sentences and queries are closer to each other. More precisely, AE improves the ROUGE-2 recall of *Ltf* by on average 11.2%. Also, AE (*Ltf*) in most cases significantly outperforms the *tf-idf* state-of-the-art term matching baselines. To investigate the semantics relevance of the extracted summary, we examined a randomly selected case study from each data set and found that the extracted sentences are mostly highly informative.

We also presented noisy AE approaches, where random noise is added to the input and output of the AE during training and testing. Using a single AE with noisy input generally reduces test performance, but using an ensemble of 5 noisy AEs restores the recall performance and is generally competitive with AE *Ltf*. Further, the case study shows that the ensemble can extract a summary that is more closely aligned to human summaries.

In future, we intend to extend our proposed model to generic summarization by clustering the AE features. A semi-supervised learning technique may be useful because of the limited availability of manually annotated data, or noisy labelling could be used with the unlabelled data. Other ensemble approaches should also be investigated to improve performance including accuracy.

References

Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009. ISSN 1935-8237.

Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*, pages 437–478. Springer, 2012.

Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541. ACM, 2006.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96. ACM, 2005.

Ziqiang Cao, Furu Wei, Li Dong, Sujian Li, and Ming Zhou. Ranking with recursive neural networks and its application to multi-document summarization. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Miguel A Carreira-Perpinan and Geoffrey E Hinton. On contrastive divergence learning. In *Proceedings of the tenth international workshop on artificial intelligence and statistics*, pages 33–40. Citeseer, 2005.

Ronan Collobert and Samy Bengio. Links between perceptrons, mlps and svms. In *Proceedings of the twenty-first international conference on Machine learning*, page 23. ACM, 2004.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011. ISSN 1532-4435.

John M Conroy and Dianne P O'leary. Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 406–407. ACM, 2001.

Hoa Trang Dang and Karolina Owczarzak. Overview of the tac 2008 update summarization task. In *Proceedings of text analysis conference*, pages 1–16, 2008.

Misha Denil, Alban Demiraj, and Nando de Freitas. Extraction of salient sentences from labelled documents. *arXiv preprint arXiv:1412.6815*, 2014.

Harold P Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.

Dumitru Erhan, Pierre-Antoine Manzagol, Yoshua Bengio, Samy Bengio, and Pascal Vincent. The difficulty of training deep architectures and the effect of unsupervised pre-training. pages 153–160, 2009.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 11:625–660, 2010.

Pierre-Etienne Genest, Fabrizio Gotti, and Yoshua Bengio. Deep learning for automatic summary scoring. In *Proceedings of the Workshop on Automatic Text Summarization*, pages 17–28, 2011.

Xavier Glorot, Yoshua Bengio, and Yann N Dauphin. Large-scale learning of embeddings with reconstruction sampling. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 945–952, 2011.

Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, pages 993–1001, 1990.

Geoffrey Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 9(1):926, 2010.

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

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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

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

Geoffrey E Hinton and Richard S Zemel. Minimizing description length in an unsupervised neural network. *Preprint*, 1997.

Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006b.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982. ISSN 0027-8424.

Khosrow Kaikhah. Text summarization using neural networks. *Faculty Publications, Texas State Universit*, 2004.

Nicolas Le Roux and Yoshua Bengio. Representational power of restricted boltzmann machines and deep belief networks. *Neural computation*, 20(6):1631–1649, 2008.

Liangda Li, Ke Zhou, Gui-Rong Xue, Hongyuan Zha, and Yong Yu. Enhancing diversity, coverage and balance for summarization through structure learning. In *Proceedings of the 18th international conference on World wide web*, pages 71–80. ACM, 2009.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8, 2004.

Elena Lloret and Manuel Palomar. Text summarisation in progress: a literature review. *Artificial Intelligence Review*, 37(1):1–41, 2012.

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

Vanessa Loza, Shibamouli Lahiri, Rada Mihalcea, and Po-Hsiang Lai. Building a dataset for summarization and keyword extraction from emails. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, 2014.

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

Inderjeet Mani and Mark T Maybury. *Advances in automatic text summarization*. MIT press, 1999.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

Ani Nenkova and Kathleen McKeown. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer, 2012.

Ani Nenkova, Rebecca Passonneau, and Kathleen McKeown. The pyramid method: Incorporating human content selection variation in summarization evaluation. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(2):4, 2007.

Jiquan Ngiam, Adam Coates, Ahbik Lahiri, Bobby Prochnow, Quoc V Le, and Andrew Y Ng. On optimization methods for deep learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 265–272, 2011.

You Ouyang, Wenjie Li, Sujian Li, and Qin Lu. Applying regression models to query-focused multi-document summarization. *Information Processing & Management*, 47(2): 227–237, 2011.

Martin F Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

Rajesh S Prasad, UV Kulkarni, and Jayashree R Prasad. Connectionist approach to generic text summarization. *World Academy of Science, Engineering and Technology*, 55, 2009.

Vahed Qazvinian, Dragomir R Radev, and Arzucan Özgür. Citation summarization through keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 895–903. Association for Computational Linguistics, 2010.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 833–840, 2011.

Ruslan Salakhutdinov and Geoffrey Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009.

Rajesh Shardan and Uday Kulkarni. Implementation and evaluation of evolutionary connectionist approaches to automated text summarization. *Journal of Computer Science*, 2010.

Nitish Srivastava and Ruslan R Salakhutdinov. Multimodal learning with deep boltzmann machines. In *Advances in neural information processing systems*, pages 2222–2230, 2012.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Krysta Marie Svore, Lucy Vanderwende, and Christopher JC Burges. Enhancing single-document summarization by combining ranknet and third-party sources. In *EMNLP-CoNLL*, pages 448–457, 2007.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

Jan Ulrich, Gabriel Murray, and Giuseppe Carenini. A publicly available annotated corpus for supervised email summarization. 2008.

Jan Ulrich, Giuseppe Carenini, Gabriel Murray, and Raymond T Ng. Regression-based summarization of email conversations. In *ICWSM*, 2009.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.

Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, 26(3):13, 2008.

Wayne Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee-Peng Lim, and Xiaoming Li. Topical keyphrase extraction from twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 379–388. Association for Computational Linguistics, 2011.

Sheng-hua Zhong, Yan Liu, Bin Li, and Jing Long. Query-oriented unsupervised multi-document summarization via deep learning model. *Expert Systems with Applications*, 42 (21):8146–8155, 2015.