Text Classification Using Local Intrinsic Dimensionality

A Minor Thesis in the Field of Text Mining

for the Degree of Master of Science in Computer Science

Peishan Li

Student Number: 905508

Supervisor: James Bailey

Credit Points: 75

Type of Project: Conventional research project

Subject Code: COMP90070

School of Computing and Information Systems

University of Melbourne

Parkville, Victoria, Australia

9 June 2019

Abstract

Local Intrinsic Dimension model is suitable for data mining and machine learning applications. We exploit LID in texts, aiming at exploring the properties of word LID and tackling some similarity-based tasks. We find that LID can represent the semantic and lexical information of words by encoding the local dimensional structure of words. Then, we propose the context-based LID, a weighting schema, which can determine the more informative words in the document. It serves as an indicator, similar to IDF. It models the interactions and relationship among words and captures the contextual information in texts. We propose a document vector model with context-based LID and apply it to text classification. The experiments demonstrate that our algorithm achieves relatively high accuracy. Moreover, Our method provides new insights into the applications of LID and the distributed representation in natural language processing.

Declaration

I certify that

- this thesis does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person where due reference is not made in the text.

- where necessary I have received clearance for this research from the University's Ethics Committee and have submitted all required data to the School

- the thesis is around 13000 words in length (excluding text in images, table, bibliographies and appendices).

Acknowledgements

I would first like to express my gratitude to my supervisor Prof. James Bailey for the insightful advice and continuous support throughout my research project. In this three-semester research project, his guidance helped me at all times, especially when I was confused or came across difficulties in my research. He also steered me in the right direction when it is needed.

I would also like to thank Dr. Sarah Monazam Erfani for her valuable suggestions during the investigation process of this research topic. My method was motivated by our discussion.

TABLE OF CONTENTS

Lists of Tables

List of Figures

1. Introduction

Text classification is an important component in many applications in the fields of natural language processing and information retrieval. In this chapter, we introduce text classification, word embedding and local intrinsic dimensionality respectively as the crucial background information. Based on the word embedding models, we propose a novel method using local intrinsic dimensionality to tackle the task of text classification.

1.1. Text Classification

Textual data refer to systematically collected material consisting of written, printed, or electronically published words, typically either purposefully written or transcribed from speech. Textual data is ubiquitous: emails, news, chats, social media, web pages, social media and more. Therefore, textual data is a rich source of information and one of the most essential components in natural language data. However, one of the most significant characteristics of the text is that text is unstructured data. Thus, processing text and extracting information and insights from text can be difficult due to its unstructured nature.

Text classification, also known as text categorisation, is one of the fundamental tasks in natural language processing. The goal is to assign predefined categories to texts. It has broad applications, such as spam detection, sentiment analysis, topic modelling, authorship attribution, native-language identification, web search, information retrieval and intent detection.

There are mainly three groups of systems and approaches to text classification. The tradictional and oldest system is ruled-based systems. By using sets of manually generated linguistic rules, the rule-based system classifies text into organised groups. A rule in the system is some patterns and semantically relevant elements, which can instruct the system to identify the appropriate category based on the text’s content. The second system is machine learning based systems. Through feature extraction and model training, a machine learning classifier is learned and can be used to classify new text. The third one is hybrid systems. Usually, a base classifier is trained and then combined with a rule-based system in order to improve the performance of the systems. Our primary focus is the machine learning based systems.

For the machine learning-based systems, the methods for text classification are different in terms of features design and extraction and machine learning classifiers. We discuss these two aspects in more details as follows.

Feature representation is a critical problem in text classification. The most relevant features for the classification are selected or constructed through some feature extraction methods. The most commonly used and simple feature extraction method is based on the bag-of-words model, such as unigram, bigram and n-gram methods. Besides these types of feature representations, more complicated feature extraction models are proposed and can reach better performance in different scenarios and contexts. For many state-of-the-art methods, dense and low-dimensional feature vectors are favoured. They can be regarded as the distributional representation of the text. Convolutional neural network, deep artificial neural network, recurrent neural network and attention mechanism are some of the advanced techniques which have been applied to obtain the effective distributional representation.

Considering the classifiers, many commonly used classifiers can be applied to this task, including decision trees classifier, SVM classifier, nearest-neighbours classifier and some probabilistic classifiers such as logistic regression and naïve Bayes. Recently, neural-network-based approaches were used and proved to be effective in text classification. In general, in a multilayer neural network system, there would be different categories of layers which are responsible for a variety of components. For example, the front layers aim at learning the representation of constituents in the text, such as words or sentences; in the middle, some pooling layers are employed to favour the more representative and important features; eventually, a softmax layer is often used as the output layer for classification.

1.2. Word Embedding

As the foundation of text representation, word representation aims at representing a word as a vector, which can be used to both compute semantic relatedness between words and feed machine learning systems as word features.

Word embeddings, also known as distributed word representation, was first proposed in 1986[1]. Word embedding involves a set of language modelling and feature learning techniques in natural language processing, where words or phrases from the vocabulary are mapped to vectors of real numbers. Compared with the original word space with one dimension per word, embedded word space is a continuous vector space with a much lower dimension.

With word embeddings, the semantic meanings of a word are encoded into a real-valued low-dimension vector. This is the primary objective of word embedding. A useful embedding provides vector representations of words such that the relationship between two vectors mirrors the linguistic relationship between the two words. The embeddings can reflect multiple degrees of similarity between words and capture both syntactic and semantic regularities.

Concerning for the methods, word embedding can be learned from count-based methods or prediction-based methods. Count models have a long and rich history. The counts of co-occurrence of words are the underlying and most important elements in count-based models. Latent Semantic Analysis(LSA)[2], Latent Dirichlet Allocation(LDA)[3], positive Pointwise Mutual Information(PPMI)[4], Local Mutual Information(LMI)[5], Singular Value Decomposition[6] and Non-negative Matrix Factorization[7] are some of the influential count-based models. As the development of neural networks, neural networks are employed to learn context-predictive word vectors. The underlying intuition is the close relatedness between the word and its context, the surrounding words. Word2Vec[8], GloVe[9], fastText[10] are some of the most popular prediction-based approaches. Generally speaking, on a variety of benchmarks for testing word vectors, overall, prediction-based methods outperform count-based methods remarkably[11].

Word similarity and word analogy are two main intrinsic evaluation tasks for word embeddings, to directly test the syntactic or semantic relationships between words. Semantic relatedness, synonym detection, and concept categorisation are some other evaluation tasks. Besides, in extrinsic evaluation, word embeddings are used as input features for a downstream task, and they can be measured by the changes in performance metrics specific to that task. Examples include part-of-speech tagging and named-entity recognition.

In a word, the use of word embeddings in an unsupervised fashion from lots of text has become a key “secret sauce” for the success of many NLP systems in recent years. Word embedding can play an important role in many applications, across tasks including named entity recognition, word sense disambiguation, part-of-speech tagging, syntactic parsing, language modelling, and knowledge extraction.

1.3. Local Intrinsic Dimensionality

In reality, multidimensional data is ubiquitous. For example, when we analyse some real-world data like real estates, we can see many inherent attributes, such as the number of rooms, the distance to shopping centre, the precise location and the suburb that it is located. In computer science, in order to make computer comprehend the data, the most commonly used method is digitisation, generating a series of numbers to describe the attributes of the data discretely. In this context, to vectorise the data is a popular way to represent multidimensional data.

In the big data era, with the explosive growth of data, we can see a considerable increase in the dimensionality of datasets. However, for most of the data mining applications, increasing representation dimensionality of data often results in the loss of performance. This is the "curses of dimensionality". However, in the high-dimensional data analysis community, there is a consensus that the data is not truly high-dimensional. Instead, the data can be projected into a subspace with lower dimension, and most of the information can be preserved. Thus, the intrinsic dimensionality is not strictly dependent on the number of features or domains. Instead, it is the characterisation of the dimensionality of the datasets.

As we discuss previously, intrinsic dimensionality can be regarded as an important indication of the complexity of the datasets. How to estimate and determine the intrinsic dimension of the datasets becomes a heated research area. Recently, many methods and models have been proposed to obtain the intrinsic dimensionality (ID). The first category of methods mainly depends on dimension reduction techniques. There are many dimension reduction approaches, which can learn the appropriate dimension of the subspace and minimise the loss of information. The dimension of this subspace can reflect the intrinsic dimensionality of the original datasets.

In addition, other approaches taking "local" properties into account are the new trends. Expansion dimension model[12] is one of these approaches to estimate the intrinsic dimensionality. It is applicable in the Euclidean space. And basically, the underlying intuition is that the dimension can be approximated with the ratios of volumes to radii, regarding ball coverage. Expansion dimension and its variants have a wide variety of applications. It can help to accelerate the similarity query and also detect the outlier. Based on the expansion model, local intrinsic dimensionality is put forward, concerning continuous random variables in extreme values.

Local intrinsic dimension(LID) model[13] further extends expansion dimension model. It assumes an underlying but unknown distribution of distances from a reference data point. Its objective is to model these distributions of distances with the help of extreme-value theory.

LID aims to quantify the local ID only considering the neighbourhood of the reference point in the data domain. We introduce the definition and formula of LID model in the next chapter.

Because LID serves as an indicator of the distance distribution of a vicinity, it is very beneficial in many similarity-based and locality-focused tasks. In addition, LID makes no assumption of the data except continuity, and this nature of LID implies its potential for broad applications in many machine learning and data mining contexts.

1.4. Our Approach

Inspired by the wide applications of LID in the fields of data mining and machine learning, we aim to employ LID in text mining. Words or phrases can be regarded as the base elements of texts. It is interesting to investigate the words using LID. The first motivation is that word embedding vectors are claimed to be able to grasp the similarity between words semantically. More importantly, LID is particularly suitable for similarity-based applications. Meanwhile, it has been noted that LID can be used to measure the complexity of the data point in terms of the local dimensional structure. Therefore, we attempt to apply LID to the datasets of word embedding vectors, seeking to explore the inherent complexity of words as well as the similarity relationships among words. We propose a new approach to quantify the similarity of word pairs, making use of the common neighbours of the word pairs and LID of words.

Except for the exploration of word LID, we intend to investigate LID in long texts, such as documents. We consider the word LID in the context of document and propose context-based LID, aiming at modelling the interactions and relationships among the words in a document. Based on our assumption, context-based LID can discriminate the more informative from the less relevant words in the document. We embed the context-based LID into the distributed representation of documents. Our propose is to implement the task of document classification with the contextual and semantic information encoded in LID.

Experimental results show that LID can imply the complexity of words and reflect some attributes of words to some extent. Our model for evaluating word similarity, which combines LID information and common neighbour intuition, outperform both some baselines and word embedding vectors in the task of word similarity. Furthermore, in the context of documents, the LID of word serves as the indication of the word relevancy to the category of documents. The experimental results show that the document vectors with context-based LID information perform well in text classification with relatively high F1-score. Compared with IDF model, context-based LID is also a powerful weighting factor which prefers the more informative words in the document.

The structure of the remaining chapters is organised as follows: in Chapter 2, we introduce the related work in the fields of text classification and local intrinsic dimensionality; we then propose our algorithms for word similarity and text classification using LID in texts in Chapter 3; Chapter 4 analyses and explains the experiment results in terms of many aspects; we make conclusions on our approaches and contributions, and plan for the future work in Chapter 5.

2. Related Work

2.1. Text Classification and Explanation

2.1.1. Text Representation

Text representation is usually the starting point for text classification. It is a method used to transform texts into numerical representations in the form of vectors. It is called the vector space model. After the original texts are converted into vectors, these vectors are usually employed to downstream processes as features of texts. Thus, text representation can also be seen as feature representation. The most frequently used method is bag-of-words model. It is a simplifying representation. In bag-of-words model, a text is represented as the bag of its words, ignoring grammar and even the order of words but keeping the frequency of words. Text representation models, such as bag-of-words, can be the foundation of many downstream NLP tasks and Information Retrieval.

In information retrieval, TFIDF (term frequency-inverse document frequency) is one of the most important techniques. The TFIDF provides a weight for each word which can measure the importance of a word to a document in a corpus. When processing text like documents in a collection, we can regard TFIDF model as the modified bag-of-words model. The TFIDF weight of words is used to replace the term frequency of words in the original bag-of-words model.

Features to be fed to the machine learning systems can be directly extracted from the text representation, such as unigrams derived from the bag-of-words model, and bigrams or n-grams from the bag of n-grams. In addition, based on the text representation, some more complex features can be processed and extracted, such as some exquisitely designed patterns based on BoW model. The main disadvantage of bag-of-word model and related feature representations is the sparsity and high dimensionality. Another disadvantage is that these conventional feature representation methods disregard the contextual information or the order of words in texts. They also cannot capture the semantics of the words. Bag of n-grams model can preserve the sequence of the words locally and capture the phrases to some extent, and it may achieve better performance and benefit from the order of the words. However, the global contextual information is lost, and the large number of features in bag of n-grams also brings other problems.

Besides, in the research of information retrieval, some dimensionality reduction methods have already been used to create low-dimensional dense vectors to represent text. Latent Semantic Indexing(LSI)[14] is a popular indexing and retrieval method to produce low-dimensional text representations using word co-occurrence. It depends on a mathematical technique called Singular Value Decomposition(SVD). It aims to project the original sparse text space into an approximate subspace which can preserve and capture the significant semantic similarity. Furthermore, there are many other feature selection methods, such as Latent Dirichlet Allocation(LDA)[3] and probabilistic Latent Semantic Analysis(pLSA)[15][14], which can select more discriminative features. With these ways, we can have feature representation with lower dimensionality.

Word embeddings provide us with an inspiration that low-dimension real-value vectors can capture a great number of information. Both syntactic and semantic relationships between words are well represented by the word vectors. Inspired by the distributed representations of words, some models are proposed to create distributed representations of sentences and documents, also known as document embeddings, aiming to capture the significant information in texts. They can be a generic text representation method for multiple downstream tasks, and due to the fixed length of the representation vectors, the vectors can also be directly used as the features or input of the machine learning algorithms. Lower dimension can bring a lot of benefits. The noise is easily eliminated and it can avoid overfitting. Low-dimension vectors represent fewer features but preserve the most relevant ones. And it is more practical and efficient for varies machine learning classifiers with fewer features.

Many approaches attempted to generate the distributed text representation with the help of word representation. One of the simple ways is that all of the word representations in the text are averaged into a text representation. Another solution is to assign a weight to each word embedding vector and aggregate them into a text representation vector[16]. Weighted average proves to be efficient and useful. These weights can be determined by some models such as IDF or learned using some machine learning methods, such as with a globally applicable weighting scheme. Document Vector through Corruption[17] includes a corruption mode and introduces a data-dependent regularization that favours informative or rare words while forcing the embeddings of common and non-discriminative ones to be close to zero. The adjustment of weighting is done during training, and the training objective and procedure is similar to that of word2vec. Some other combinations of word vectors are also practical. For instance, Max or Min integration[16] refers to that for each dimension of vectors, we take the maximum or minimum across all word embeddings, as the value of the document vectors at that dimension.

2.1.2. Text Classification Approaches

In this section, we introduce some state-of-the-art methods for text classification. Most of them made full use of neural networks but varied in terms of the types of neural networks and learning objectives. Some of them also reformed the distributed representation of documents.

Convolutional Neural Networks

Conneau et al.[18] first used very deep convolutional neural networks to solve the task of text classification. Convolutional neural networks work well for many computer vision and NLP tasks, especially for the data with inherent hierarchical properties. Considering texts, we can see the internal compositional structure of text: characters combine to form words; words form phrases and sentences; sentences form documents. In this work, they used very deep convolutional networks with 29 layers to learn the representations of sentences. Because of the great depth of the ConvNets, information of characters is captured to build the sentence by bottom-up approaches. The sentence representation serves as the hidden layer of the convolutional networks. Using k-max-pooling, the representation output is first down-sampled to extract the k most important features. After that, some fully connected layers and softmax layer are applied as classifiers in order to implement the classification of categories.

Recurrent Neural Network

Lai et al.[19] introduced a recurrent convolutional neural network for the task of text classification. Different from the generic word embedding approaches, the word representation is learned using a bi-directional recurrent network. In this way, the contextual information is captured as far as possible and integrated into the word vectors during the learning. A more precise word meaning in the document is well represented by the word vector. Eventually, a max-pooling layer is employed in order to determine and select the key components in texts. Compared with the window-based methods, global information is preserved and the noise can be remarkably reduced.

Distributed Representation of Documents

Joulin et al.[20] put forward a new word embedding methods, taking into account the subword information. They think the character features are significant and can enrich the word vectors. The bag of n-gram characters model is built and the subword character vectors are learned. The representation of the word is the sum of the n-gram characters vectors. Based on the fastText word embedding methods, an efficient and simple baseline for text classification is proposed. The word vectors are aggregated to form the text representation. In the model architecture, the text vectors serve as a hidden layer and then the hierarchical softmax classifier is trained which is fed with the hidden text representation and the n-gram features.

Le et al.[21] proposed Paragraph Vector for distributed representations of sentences and documents. Inspired by the CBOW model, they aimed to utilize the surrounding context word vectors and a specific paragraph vector to predict the centre word. It is an unsupervised algorithm and the word vectors are shared among all the documents. Another version of the Paragraph Vector ignores the word order and it is named distributed bag of words(DBOW). For the text classification, the original paragraph vector and PV-DBOW are concatenated as the feature vectors for text classification.

Hierarchical Attention Networks

Yang et al.[22] introduced a hierarchical attention network for document classification. The intuition underlying the method is that the document is comprised of smaller elements hierarchically, from words to sentences bottom-up. In addition, not all parts of the document are equally important and relevant, and thus the more significant content should be stressed and contribute more to the classification task. From this motivation, a two-level hierarchical architecture is constructed, including the word encoder and the sentence encoder. This architecture mirrors the natural hierarchical structure of documents. Moreover, two attention mechanisms are employed to differentiate the more or less informative constituents, in word-level and then in sentence-level.

2.2. LID and its Applications

2.2.1. LID and Explanation

In data mining, the estimates of the complexity of datasets are useful and important. Because they can be used in the design and analysis of some data mining algorithms, such as the algorithms for search, classification, clustering and outlier detection. But the number of attributes or features is not the only factor to measure the complexity of datasets. Therefore, many theories and approaches are proposed to measure the complexity of datasets and characterize the data with intrinsic dimensionality. Basically, intrinsic dimensionality is the minimum number of dimensions needed to represent the dataset without information loss.

Over the past few decades, a great number of models have been put forward in order to measure the intrinsic dimensionality of data sets. Examples include famous Principle Component Analysis(PCA) and its variants[23]. In addition, some fractal methods, like Correlation Dimension(CD)[24], use the space-filling capacity of the data. Graph-based methods[25] attempted to evaluate the ID combining the k-Nearest-Neighbours and the density of the data. Moreover, some topological methods[26] utilized some local samples to estimate the basic dimension of the data space. But they all focused on the global property of ID regarding the whole datasets.

The characteristic of local neighbourhoods can be hugely different from the properties of the global dataset. Local structure of the data point of interest is especially notable when similarity queries are operated. As we discussed previously, most of the ID estimation approaches are involved in the "global" characteristic, treating the dataset as a whole but neglecting the individual data points.

Karger and Ruhl[12] proposed the Expansion Dimension to describe the intrinsic dimensionality of the datasets. It is similar to another measure of intrinsic dimensionality, the doubling dimension, which is based on the ball coverage. In an m-dimensional Euclidean space, when we consider two balls with different radii, the dimension m can be deduced from the ratios of volumes. But this expansion dimension method can only be applied to Euclidean spaces.

Houle et al.[27] then proposed Generalized Expansion Dimension(GED) based on the original expansion dimension model. It originally seeks to analyse the similarity search indices in

Euclidean space. Then the generalized expansion dimension is extended to multiple spaces with different similarity measures, such as cosine similarity(vector angle). Besides estimations of intrinsic dimensionality throughout the whole datasets, generalized expansion dimension model can be applied within local neighbourhoods to analyse intrinsic dimension as well. The minimum neighbour distance(MiND)[28] is another measure of ID concentrating on "local" property. It is motivated by the bias in some ID estimates methods, which results in the underestimated effects. It employed the manifold embedded learning to reduce the dimensionality and exploit the statistical characteristics of manifold neighbourhoods.

Local intrinsic dimension(LID)[13] can be regarded as an extension of Expansion Dimension in a statistical setting with continuous random variables. This local model extends the GED model with the smooth functions and conditions, and it is established within the extreme value theory(EVT) framework. LID aims to quantify the local ID regarding the neighbourhood of a point of interest in a feature space. Another extension is that LID makes an assumption about the underlying continuous probability distributions in terms of the distance from the point of interest. It is unknown distribution and LID assumes the smoothness of the distribution, which refers to that the distribution is continuously differentiable over ranges of interest.

LID can be treated as the equivalent of the indiscriminability of the distance measure. As the radii increase, the probability measure changes largely relatively and the growth rate of the cumulative distribution function is large. In this case, it implies high indiscriminability of the random variable. When we are concerned with the measurement of error or the exploration of neighbourhood sets, the relative growth rate is more important than the absolute increase. Therefore, the indiscriminability is crucial in some tasks, such as similarity search and error measurement. Because LID is a perfect indicator of indiscriminability, LID can measure the cost of expanding the search radius and performing some similarity-based queries within a local neighbourhood.

We provide the definitions of indiscriminability and LID as follows. *Figure 1* shows the local neighbourhoods of two random distance variables and we can observe the large difference of indiscriminability between these two variables.

Definition 1: Let X be an absolutely continuous random distance variable. For any x such that $\mathrm{F}_X(x) > 0$, the indiscriminability of X at x is given by

$$InDiscrX(x) = \lim_{\epsilon \to 0^+} \frac{\mathrm{F}_X((1+\epsilon)x) - \mathrm{F}_X(x)}{\epsilon \cdot \mathrm{F}_X(x)}$$

wherever the limit exists.

Definition 2: The *local intrinsic dimensionality* of X at distance r is

$$IntrDimX(r) = \lim_{\epsilon \to 0^+} \frac{ln(\mathrm{F}((1+\epsilon)r)/\mathrm{F}(r))}{ln(1+\epsilon)}$$

wherever the limit exists.

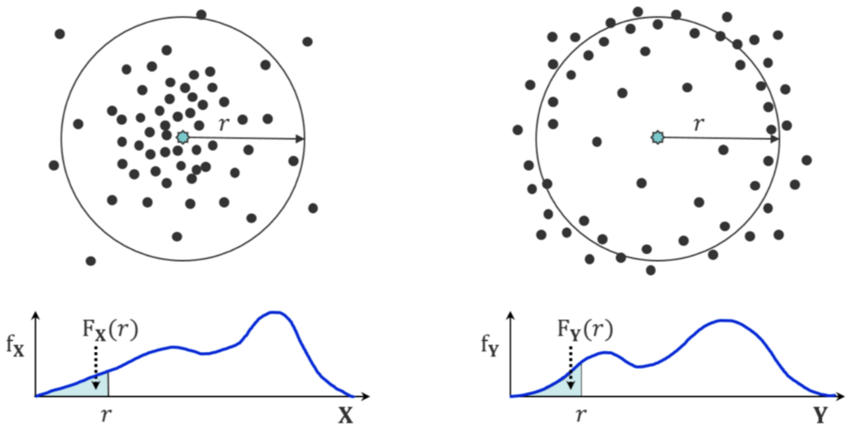


Figure 1. Indiscriminability of random distance variables X and Y

Many estimators of LID have been proposed in terms of extreme value theory. Maximum Likelihood Estimation(MLE) is the most popular one. Hill estimator is a well-known estimator which is appropriate for the scaling exponent of a power-law tail distribution. The Hill estimator for LID is shown in *Formula 1*. The MLE model can guarantee the regularity conditions and the consistency of the estimation. And the MLE estimator is usually efficient with acceptable accuracy. Moreover, there are some other types of estimation methods, such as moment-based estimators[29] and the estimator based on regularly varying functions.

Formula 1: Hill estimator of LID is given by

$$\widehat{ID}_{\mathrm{F}}(0) = -\left(\frac{1}{k}\sum_{i=1}^{k} ln\frac{r_i}{r_k}\right)^{-1}$$

where r_i is the i-th smallest distance in the sample neighbourhood.

2.2.2. Applications of LID

Local intrinsic dimensionality has broad applications in data mining. Particularly, it can be helpful and beneficial in clustering, classification, outlier detection, similarity search in many contexts. Besides data mining, LID has been proven to be effective and useful in some other applications, including machine learning and database.

The first kind of applications is about similarity query. Casanova et al.[30] introduced an approximation approach to reverse k-Nearest neighbour(RkNN) based on LID. LID can also be employed to improve the accuracy of the k-Nearest-Neighbours graph.

Moreover, LID was first proposed aiming to solve many data mining problems, but it turned out that it is closely associated with the theory of extreme values. Romano et al.[31] theoretical proved the connection between the intrinsic dimensionality and information theory and proposed an approach using LID to measure the dependency among the variables in the dataset. Dependency measurement of variables is of great importance in pattern recognition. They identified the variables with low intrinsic dimensions with extreme-value theory.

LID model concerns the local subspace and its dimensional structure. It can be useful to characterize the subspaces of different types of data points. Therefore, local intrinsic dimensionality can be an indication of outliers. Brunken et al.[32] introduced a generic method for evaluating local outliers in high-dimensional data with the help of LID. They tested this intrinsic dimensional outlier detection method on four datasets, compared with some start-of-the-art outlier detection approaches. Ma et al.[33] utilized LID to characterize the special properties of adversarial examples in Deep Neural Network(DNN). It motivates the new direction for the detection of adversarial regions. Besides, another endeavour to understand the adversarial samples in machine learning systems also employed LID[34]. After the investigation, they systematically proved the impact of LID on adversarial perturbation.

Besides the applications in similarity search, variable dependency and outlier detection, LID has also been employed in some cutting-edge fields. Houle et al.[35] proposed LID-Fingerprint, a binary fingerprinting technique for high-dimensional data. It aims to represent the data in a more compact way compared with the original data. The LID-Fingerprint is very promising for secure and efficient similarity queries. Ma et al.[36] utilized LID as a tool to assess the learning behaviour of a deep neural network. From the perspective of dimensionality, they developed a new dimensionality-driven learning strategy which is highly tolerant of the noisy labels. LID is exploited to capture the characteristic of the deep representation subspaces and monitor the learning process.

3. LID in Texts

3.1. LID of Words

Local intrinsic dimensionality can provide a local view of the dimensional structure of the data. We assume that local intrinsic dimensionality can be an indicator of the complexity of words. But what is the complexity of words? We select and define a few attributes of words in order to systematically analyse the effectiveness of LID of words.

3.1.1. Definition of Word LID

In the theory of intrinsic dimensionality, classical expansion models measure the rate of growth in the number of data objects encountered, as the distance from the reference sample increases. Compared with other methods for ID estimates, one of the distinguishing features of expansion models is that they highlight the local property of ID. They usually involve the local neighbourhoods, such as k-Nearest-neighbours, and concern individual objects instead of the global dataset. With respect to LID, in the vector spaces, given a neighbourhood, it is fast and efficient to use Hill estimator to calculate the local intrinsic dimension. In this way, the LID of the certain data point can be computed according to its k-nearest neighbour distances.

We project the words into a low-dimensional vector space using word embeddings. In the datasets of the word vectors, we apply Hill estimator to the calculation of word LID. The formula 2 below gives the definition of word LID.

Formula 2: Hill estimator of word LID is given by

$$\widehat{ID}_{\mathrm{F}}(w_c) = -\left(\frac{1}{k}\sum_{i=1}^{k} ln\frac{r_{w_i}}{r_{w_k}}\right)^{-1}$$

where w_c is the word vector of interest, r_{w_i} is the distance from w_c to its i-th nearest neighbour, and r_{w_k} is the largest distance in the sample neighbourhood of w_c.

3.1.2. Word Attributes

We define three word attributes, including word frequency, word sense and word class. In the table below we explain the attributes of words briefly.

Word Frequency	Frequency of occurrence in a text corpus
Word Senses	The different meanings that a word has. It can depend on the “synsets” in WordNet.
Word Class (POS)	Part-of-speech: Nouns, verbs, adjectives, adverbs, prepositions, determiners, pronouns, conjunctions and modals.

Table 1. Word attributes and explanations

We first introduce some important resources which help us describe the attributes of words. There are many manually constructed resources about word semantics such as thesauri, lexicons and ontologies. WordNet is an important lexical database of English but also supports most major languages globally. For English WordNet, there are around 120000 nouns, 12000 verbs, 21000 adjectives and 4000 adverbs, etc. The nodes of WordNet are not words, but meanings, which are called synonyms or synsets. In WordNet, words are grouped into sets of synonyms based on their meanings. The words which denote the same concept and can be interchanged in many contexts belong to the same synset. However, because many of the English words have more than one meanings, a word can belong to multiple synsets. On the other hand, in WordNet, connections between nodes are lexical relations. For the relations among synsets, different kinds of relationships are encoded, including super-subordinate relation, part-whole relation, opposite relation, troponym relation and so on.

NLTK corpora is another resource that we use for the count of word frequency. NLTK corpora, from Natural Language Toolkit(NLTK), are collected and built with a variety of human language corpora. Totally NLTK corpora include 107 corpora covering most of the tasks in natural language processing. These corpora are usually designed and built for some particular tasks in different formats. NLTK also provides a large number of useful tools for text processing.

With the help of WordNet, we can use the synsets to define the word senses. Considering a particular word, the number of word senses is the number of the synsets that the word belongs to in WordNet. With some text corpus, we can count the frequency of occurrence for each word.

We use NLTK as text corpus to count the word frequency. Word class is also known as lexical category or part of speech. In grammar, we can define the category of words according to their similar grammatical properties, especially syntactic property. We can also use WordNet to determine the word class for each word in the dataset. However, many words belong to multiple classes. For instance, “like” can be a preposition or a verb. We can detect the correct word class from the contexts. But when we analyse words without contexts, WordNet can help us to find the primary class for each word separately using statistic data. In details, for the words which belong to more than one synsets, WordNet can determine the primary sense of the word with statistical data and then we can infer the class of that primary sense(synset).

We suppose that there exist some relationships between the word LID and these attributes of words. We expect diverse distributions of LID in different subsets of words which are grouped by the word attributes. We conduct some data analytics and experiments about LID and word attributes, and the details are provided in the next chapter. The figure below shows two sample words within different ranges of LID and we can compare them in terms of word frequency, word class and word senses. We can also see the various local structures of the neighbourhoods. To simplify the visualisation of the local structure, we just use $k = 10$.

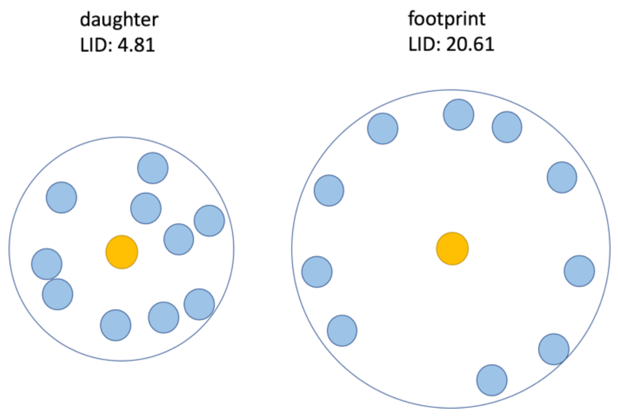


Figure 2. Two examples of word LID

3.1.3. Word Similarity with LID

Inspired by common sense in social networks, that two persons are likely to become friends when they have many common friends, we attempted to measure the similarity of two words depending on their common nearest neighbours. We assume that if two nodes are similar, when we look into their local neighbourhood respectively, we can observe similar local dimensional structures and the two nodes may locally have several shared nearest neighbours. To be more specific, the intersection of the k-Nearest-Neighbours sets of two nodes is our main concern. One simple strategy is to measure the word similarities by the number of words in their k-NN intersection sets. The more common neighbour that two words share considering a certain range of local neighbourhoods, such as 100-NN, the more similar these two words are.

Another motivation in social networks is that the significance of each shared friends is different. For instance, a common friend who doesn't have a lot of friends is very important to link prediction. To benefit from the properties of a common friend, a weight is usually assigned to each common friend. The weight can be determined by the distance, the in or out degrees. Similarly, we evaluate the similarity of word pairs based on the common neighbours in the word vector space. And we utilise word LID as the weight of each common neighbours. The estimate of word similarity using LID is defined as follows.

Formula 3: the evaluation of word similarity based on LID-weighted sum of common neighbours is given by

$$E_{sim} = \sum_{w_e \in S_x \cap S_y} \widehat{ID}_{\mathrm{F}}(w_e)$$

Where x and y are the two words we are concerned with, S_x and S_y are the corresponding k-NN sets of x and y, and as we defined previously, $\widehat{ID}_{\mathrm{F}}(w_e)$ is the LID of the common-neighbour word w_e.

The figure below illustrates the shared neighbours of the word pairs "computer" and "internet", when we use $k = 10$. The gold standard for this pair is 7.58. And the radius of the circle reflects the LID of the words in the neighbourhoods. Besides, the figure displays the vicinities of another word pairs with lower relatedness, "tiger" and "organism", and the similarity score is 4.77. We can easily differentiate these two cases and we can probably deduce that our estimate of word similarity using LID is reasonable.

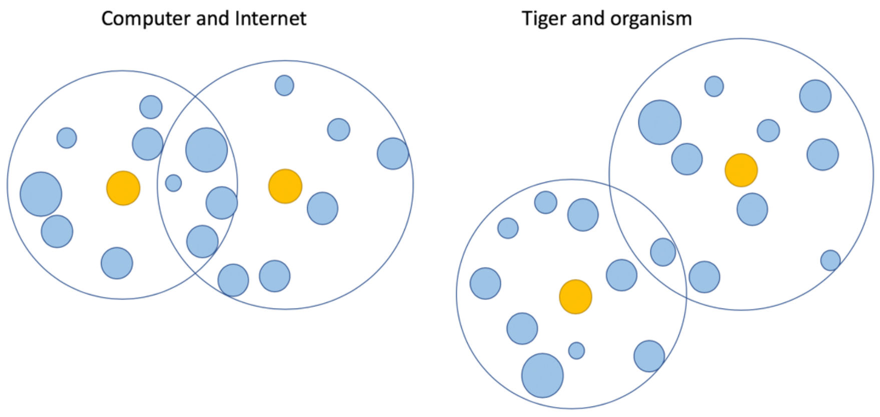


Figure 3. Local neighbourhoods of two word pairs

3.2. LID in Texts

Considering words in the text, we assume that LID can help to better capture some semantic information, after doing some systematic investigation of LID of words. The local dimensional structures of words are modelled and preserved with the LID. Besides, since the estimation of LID is restricted to a neighbourhood around the word of interest, LID takes the correlation and interaction among the words into consideration. Therefore, LID can be an indication of the relation between the centred word and the context. We assume that LID can well represent the importance of the word of interest in a document. We want to use LID as additional features or factors to improve the result of text classification.

3.2.1. Distributed representation of document

As we discussed in the literature review, there exist many approaches to distributed representation of document. In order to encode the LID of words into the document vectors, we first generate a dictionary of word LID. After that, each term in the document is assigned a weight according to its LID. The LID of word would vary when we use a different range of surroundings to calculate it and we will provide more details in the next part. All the word vectors in a document are weighted aggregated into a document vector. It is a distribution representation of document with low dimension. *Figure 4* demonstrates the process to create the document vector by summing up all the word vectors with a weighting schema.

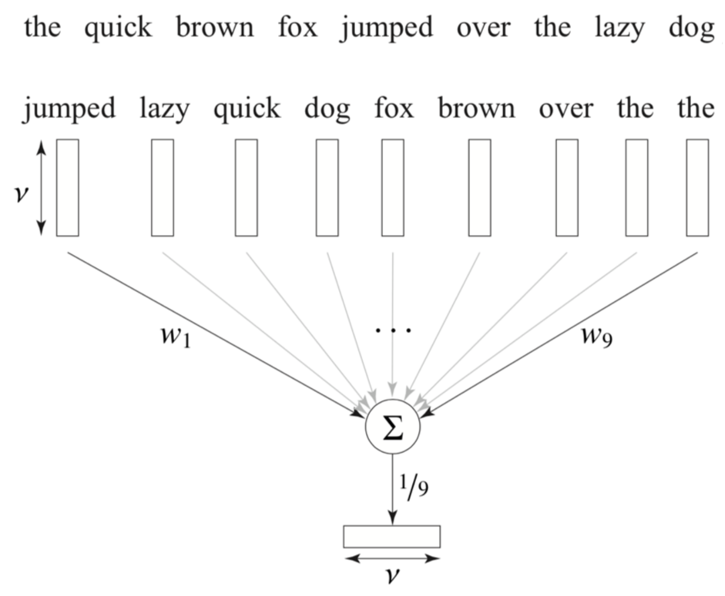

the quick brown fox jumped over the lazy dog
jumped lazy quick dog fox brown over the the
v
w_1
. . .
w_9
Σ
$^1/_9$
v

Figure 4. Aggregation of word vectors to generate document vectors

After obtaining the document vectors, the document vectors can serve as the features and we can apply different machine learning classifiers, such as SVM, random forests and neural networks.

We can build the vocabulary dictionary for the whole corpora. Instead of using some generic word lists from WordNet or some pre-trained word vectors, we are concerned about the document corpus itself. We count all the words which occur at least once in the corpus and construct the vocabulary list for the document classification dataset, like 20NewsGroups.

3.2.2. Context-based LID

Similar to what we do to investigate the generic LID of words, we can compute the LID of each word regarding the full word lists of the corpus. The whole word lists of the document collections are the global surroundings when computing LID. However, in the task of document classification, we are more interested in the importance of a word in the context of the document. Rather than measuring the importance of the word among all the words in the corpus, we should emphasise the most relevant and informative words among all the words in a specific document. Using a single document as the global context to calculate the LID is reasonable and practical. Limited in a single document, the k-nearest-neighbours of each word are easily identified and then the context-based LID is computed. In this way, the contextual information is preserved, and we can model the interactions and similarity among the words in the same document, not just the occurrence of words in isolation.

The context-based LID can help to determine the most relevant words when classifying the documents. The informative words are stressed, which is similar to the TFIDF model. TFIDF model involves the relationship between terms and documents and the significance of the terms in the surroundings of the whole document collection. The context-based LID aims to capture the relationship between the word of interest, the "centred word", and the surrounding words in a certain context. For instance, if the centred word is informative in the document with the category focus, the distances between the centred word and all the other words in the document should be large to some extent. This type of centred word has higher LID than the non-informative and common words in the document. Its word vector would be favoured when being aggregated into the document vector due to the high LID as weight.

Figure 5 displays an example of two words with their k-Nearest-Neighbours in a sample document. The context-based LID for these two words are shown as well. We can infer that "microsystems" is more informative in this document than "use" due to higher context-based LID. The category of the document where these two words are located in "computer".

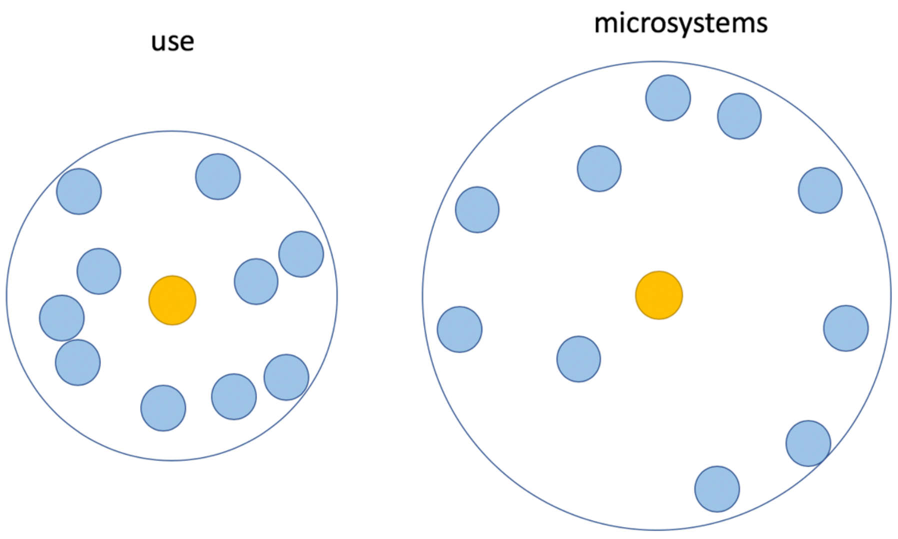


Figure 5. Two examples of context-based LID and corresponding local neighbourhoods

In addition, we can define different sizes of context. By default, we choose the whole document as the context. The other options include a sentence, a paragraph, a subset of documents as context.

4. Experiments

4.1. Investigation in Word LID

We do some investigation in word LID with respect to the probability distribution, mean, correlation and some statistical summary tools. We also evaluate the effectiveness of word LID by doing experiments in the task word similarity.

4.1.1. Datasets and Resources

Pre-trained Word Vectors

In order to use LID to explore textual data, the first step is to vectorize the words. The most influential and widely used pre-trained word vectors include the word embeddings from word2vec[8], fastText[10] and GloVe[9]. Word vectors are usually learned by unsupervised methods with large-scale text corpus.

Table 2 describes the details about the pre-trained word vectors from the three most popular word embedding models. These pre-trained word vectors are different in terms of the dimension, training sets and the size of the vocabulary.

	dimension	Training set	Vocabulary size
Word2vec	300	Google News dataset (about 100 billion words)	3M
GloVe	50/100/200/300	Wikipedia 2014 + Gigaword 5 (6 billion tokens)	400K
	300	Common Crawl (42 billion token)	1.9M
	25/50/100/200	Twitter (2B tweets, 27B tokens)	1.2M
fastText	300	Wikipedia 2017, UMBC webbase corpus and	1M

		statmt.org news dataset (16 billion tokens)	
	300	Common Crawl (600 billion tokens)	2M

Table 2. Details about pre-trained word vectors

For the investigation of LID characteristics, we used pre-trained word vectors from GloVe model, and we simply used 50-dimension word vectors to calculate LID. We intend to get some insights and inspiration from the exploration. Thus, we don't try different pre-trained word vectors and other options of dimensions.

However, for the task of word similarity, we attempt to exploit diverse pre-trained word vectors and dimensions of word vectors. Moreover, we use all the word vectors in the pre-trained vectors to compute word LID in the pre-trained vectors for this task.

25kWordLists

In order to explore the properties of word LID, we first need to construct a word list using some dictionaries. However, because the pre-trained vectors are learned with huge amount of online textual data such as Wikipedia and Google News, each unique token or term is regarded as a word and the vocabulary size is huge. The huge number of terms is impractical for the experiments and data analysis. Therefore, we filtered out the non-English and rare words with the help of some English dictionary resources.

WordNet can be regarded as a combination of dictionary and thesaurus. We extract the English words from WordNet and do some data pre-processing to build the word lists. Our word lists contain 25077 words in total. We name this word list 25kWordLists. The words in the word lists are all frequently used English words. For each word in the word lists, we utilize some tools to generate the corresponding attributes, including word frequency, word senses and word class. We use this word list for the generic exploration of LID characteristics with respect to some word attributes.

WordSimilarity-353

The WordSimilarity-353 includes two sets of English word pairs along with the score indicating the relatedness and similarity between the two words. There are 353 word pairs in total. These scores were generated based on human judgements, which can be treated as the "gold standard". This dataset is the most popular one for the task word similarity.

4.1.2. Discovery of Word LID

In terms of the categories of words (part-of-speech tagging), generally, we can observe that different categories of words have different distributions of LID. LID is also an indicator of word frequency. Frequently used words tend to have lower LID than those rare words. In addition, regarding word sense, the word with unique meaning is more likely to have low LID, such as months and family members. Based on some data analytics, we can conclude that to some extent, LID can represent some interesting lexical information of words. We would explain with some examples regarding different aspects.

Distribution of LID

Figure 6 shows the distribution of the LIDs of words and we can claim that it obeys normal distribution. The word LID less than 5.0 or greater than 25 is rare. Most of the word LID fall within the interval from 10.0 to 20.0.

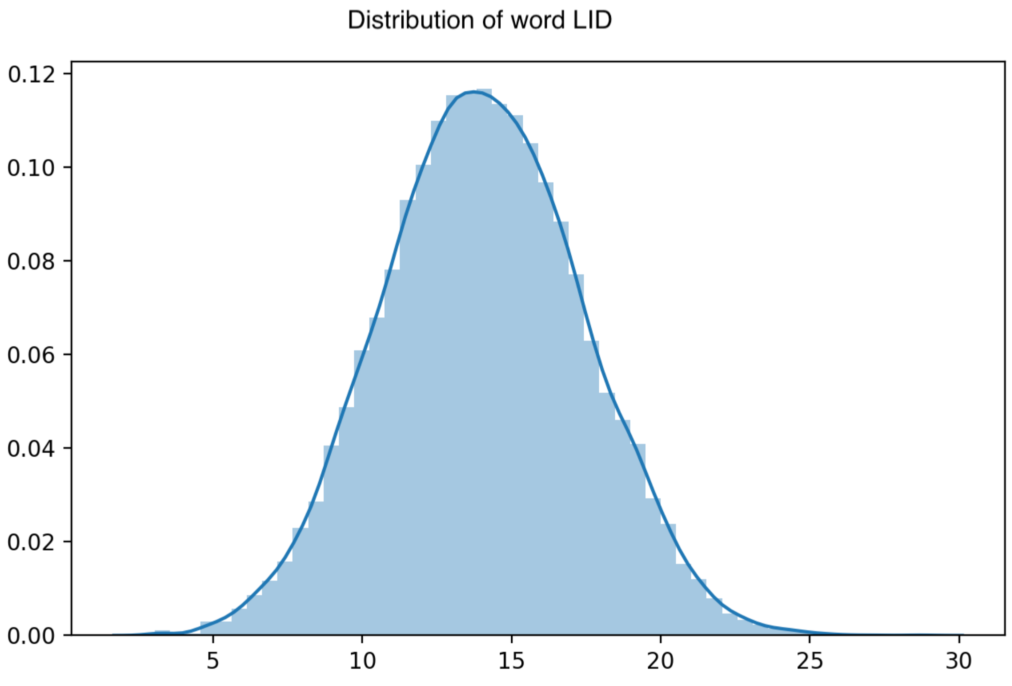


Figure 6. Overall Distribution of LID

Word Frequency

We attempt to find the relationship between LID of words and the word frequency. Words are grouped into different bins based on their frequency of occurrence in the corpus. The group of words with higher frequency tends to have a lower mean of LID.

In natural language processing, stop words usually refer to the most common words in the language. The LID of the stop words are all very small.

In addition, we try to use Spearman's rank correlation coefficient to model the relationship between word LID and word frequency, and it is approximately -0.197. We can deduce a negative correlation between these two factors.

Word Frequency	LID(mean)
more than 25000	7.668
5000-25000	8.427
4000-5000	8.832
3000-4000	9.76
2000-3000	9.108
1000-2000	10.639
0-1000	14.097

Table 3. Mean of LID in different word frequency groups

Word Senses

Regarding word sense, the word with unique meaning is more likely to have low LID. Some examples from the set of words with the LID lower than 5 are visualized in *Table 4*. Most of them describe the concrete and unambiguous concepts in English, such as the names of months, numbers, day of week and family members.

When we look into the vectors of these words with the lowest LID, we can observe the apparent clusters of words. For instance, the words describing the months in a year assemble to form a cluster. In a certain cluster, all the words belong to a specific "semantic field". In linguistics, "semantic field" or "semantic domain" refers to the set of words grouped semantically by meaning. All the words in a semantic field share common semantic property and they are all used to describe the same general phenomenon.

Semantic field	words
Months	October, June
number	Fifteen, seven
Day of week	Tuesday, Wednesday
family	Daughter, niece
Musical instrument	Cello, saxophone

Table 4. Examples of words and corresponding semantic fields with low LID

Whether the meaning of the word is unambiguous can be answered with the help of WordNet. We can use the number of “synsets” that a word has in WordNet to quantify the complexity and ambiguity of the word senses. The words with low LID are more likely to have only one or two “synsets”.

Some special cases are also interesting and should be emphasised in our investigation. We analysed all the words with exact one synset in WordNet and observed their corresponding LID. We found out some words with one word sense but very high LID. It is easy to conclude that these special samples are all some infrequent words with low word frequency. These are some instances, flay, topnotch and prewar.

Word Class

We visualize the distribution of LID in different lexical categories. We divided the word lists into subsets according to lexical categories. Most of the words belong to the Noun class. It is obvious that the LID still obeys normal distribution in every word class. Besides, the relationship between LID and word frequency remains in all subsets.

4.1.3. Word Similarity

Based on the estimation model of word similarity we proposed in the previous chapter, we use the sum of word LID in the intersection set of two k-NN sets to evaluate the similarity between the word pair. We conduct the experiment on the WordSimilarity-353 dataset. To summarise, our method outperforms both some traditional baselines and the word embedding vectors.

Baselines

(1) Knowledge-based and corpus-based

The classical approaches to measuring the semantic relatedness can be divided into two categories, the knowledge-based and the corpus-based. The knowledge-based ones rely on lexical taxonomies like WordNet, while the corpus-based methods make use of some corpus resources to generate probabilistic semantic models from the raw text statistically. We list some of these approaches as follows and use them as baselines in our experiments. But because these algorithms are not highly related to our investigation in LID, we will not provide too much details. For the experiment results of these baselines, we refer to Hassan and Mihalcea's study in 2011[38].

Knowledge-based measures: L&C, WNE, J&C, L&C, H&S, Resnik, ROCET

Corpus-based measures: Lin, LSA

(2) Word vectors with cosine similarity

We can easily utilize word vectors from some word embedding models, also known as the distributional representation of word, to measure the similarity of word with cosine similarity. We contrast our method with its corresponding word embedding model which is used to generate the pre-trained vectors. For instance, if we use fastText pre-trained vectors to create word LID, we will compare our approach with the cosine similarity of fastText word vectors. However, unlike word vectors, it is hard to directly use LID to compare the similarity between words.

(3) State-of-the-art methods

We also compare our algorithm with some state-of-the-art approaches in this research field. We briefly introduce some advanced approaches that we use for comparisons as follows.

ConceptNet Numberbatch[39]: It was developed by Speer et al. in 2017 and it is a hybrid method combining knowledge from taxonomies and patterns from raw text.

Multi-lingual SSA[40]: It was proposed by Hassan et al. in 2011, which is corpus-based.

ESA[41]: It is a corpus-based method proposed by Gabrilovich and Markovitch in 2007.

Experiment Results and Analysis

According to the categories of approaches above, our method using word embedding vectors and their LID should be a corpus-based method, since the word vectors are learned through unsupervised learning with massive amount of corpus data.

We explain some configurations of the experiments in this paragraph. The LID of each word is dependent on the distances between word vectors. Thus, it is necessary to normalize all the word vectors when pre-processing the data. To compute the LID of words, the range of the neighbourhoods is a significant factor. By Hill Maximum Likelihood Estimation method, the approximation of LID is computed with the distances from the reference point to its k-nearest neighbours. Therefore, the number of nearest neighbours that are taken into account can identify the locality. We denote k as the number of nearest neighbours, and it is a hyperparameter in our algorithm. In the experiments, we use k = 100 by default. The distance function that we consider when measure the distance among vectors is angular cosine distance. For the choice of pre-trained word vectors, we exploit fastText model with 300 dimensions here, and thus the word vectors consine similarity is also based on fastText.

For the evaluation metrics, we utilise Spearman's rank correlation coefficient(Spearman's rho) and Pearson correlation coefficient(Pearson's r) to evaluate our method and the baselines. As we introduced in the previous section, there exist gold standard scores in the WordSimilarity-353 dataset. We simply regard the similarity scores that our method produces and the gold standard score as two variables. And then we assess the relationship between them with Spearman's rho and Pearson's r respectively. The experimental results are shown in *Table 5*.

Approach	Category	Spearman's rho	Pearson's r
L&C	Knowledge-based	0.302	0.356
WNE	Knowledge-based	0.305	0.271
J&C	Knowledge-based	0.318	0.354
L&C	Knowledge-based	0.348	0.341
H&S	Knowledge-based	0.302	0.356
Lin	Corpus-based	0.348	0.357
Resnik	Knowledge-based	0.353	0.365

ROGET	Knowledge-based	0.415	0.536
LSA	Corpus-based	0.581	0.563
Word Vector	Corpus-based	0.661	**0.684**
kNN + LID	Corpus-based	**0.695**	0.497
ConceptNet Numberbatch	Hybrid	**0.828**	N/A
Multi-lingual SSA	Corpus-based	0.713	0.614
ESA	Corpus-based	0.748	0.503

Table 5.Experiment results of word similarity

From the experimental results above, in comparisons with some conventional baselines, our algorithm outperforms most of the baselines by a remarkable margin, especially when compared with some knowledge-based methods. Considering Spearman's rho, our algorithm achieves a better result than all the traditional baselines and word vectors. However, in terms of Pearson's r, word vectors measure outperforms all the other measures significantly, including the state-of-the-art.

Contrast our measure with the state-of-the-art methods, even our method cannot outperform them, the margin is not large. The accuracy of our method is acceptably high. Moreover, after obtaining the word LID in the word lists in pre-trained vectors dataset, it is cost-effective to produce the measure of word similarity by our algorithm. But the computation complexity is also very low for the measure by word vector cosine similarity.

For the knowledge-based approaches, the two metrics, Spearman's rho and Pearson's r give similar values. We can observe a similar situation when we use word vectors cosine similarity to measure word similarity. However, the Spearman's rho is much higher than Pearson's r for our measure. The Spearman's rho is less sensitive than the Pearson correlation to some strong outliers, because Spearman's rho limits the outlier to the value of its rank. When we look into the actual values of our measure of word similarity, we can find that some word pairs with little relatedness have no common neighbours at all. This kind of cases is reasonable. For these strong outliers, the cosine similarity of two vectors is not too close to zero, while the value of our measure is exactly zero. This is one of the disadvantages of our algorithm. But we can rely on Spearman's rho to avoid the effects of outliers.

The word embedding vectors can capture the semantic and syntactic information about words by the training process with huge corpus data. Our method depends on these underlying similarity relationships of word vectors as well. However, the main difference is that cosine similarity is the direct measure, whereas our model is an indirect measure of similarity between two entities by counting their common neighbours. LID serves as a weighting schema, which helps us to select the more important common neighbours. The common-neighbour words with higher LID make more contribution to the measure of word similarity.

Based on the analysis of our experiment, we can deduce that LID can imply the importance of a word when it is located in the reference point's local neighbourhood. LID is a powerful sign when we conduct similarity queries. LID can reflect some useful semantic information of words.

Hyperparameters

(1) Distance functions

The local intrinsic dimension is sensitive to the similarity function. Because locality is the key in LID, when we identify the nearest neighbours of the reference point, the definition of the distance between two data points is a significant premise. Our problem is discussed in the vector spaces, cosine similarity and Euclidean distance are two of the widely used distance metrics. In fact, NMSLIB library supports multiple distance functions, including Lp-norms distances, Manhattan distance and Chebyshev distance.

We mainly do experiments to compare Euclidean distanceL_2 and angular cosine distance in the task. We keep other hyperparameters and experiment configuration the same to evaluate the effect of distance functions. We expected that we can see a difference in evaluation metrics when using different distance functions. However, the results are just slightly different. It implies that in the vector spaces of word embedding vectors, these two distance measures are almost the same.

(2) Pre-trained word vectors and dimensions

Word embedding vectors serve as the fundamental blocks in our systems, so it is reasonable to examine the effect of word vectors by choosing different word embedding models. We attempt to apply Word2vec, GloVe and fastText respectively. The results of different pre-trained vectors are shown in below table. We can see that the fastText outperforms other two word embedding models. Thus, we utilise fastText to compute LID in the word similarity experiment mentioned above.

As for the dimensions of the vectors, we expect that when we use a higher dimension of word vectors, more semantic information can be embedded into the vectors and we can a obtain better presentation of LID. We tried 50, 100, 200 and 300 with GloVe model. In general, based on the experiment results, the higher dimension of the word vectors, the more accurate measure we can make. The reason is that by appropriately increasing the dimension of the distributed representations, more semantic and contextual information is captured.

Pre-trained vectors	Dimensions	Spearman's rho	Pearson's r
fastText	300	**0.695**	**0.497**
GloVe	50	0.522	
	100	0.547	0.461
	200	0.569	0.479
	300	0.592	0.486
Word2Vec	300	0.603	0.472

Table 6. Experiment results of word similarity with different word vectors and dimensions

(3) k in k-NN

Besides, the value of k in k-Nearest-Neighbours is another important hyper-parameter. We tried various values for k in the experiments, $k = 25, 50, 100, 200$. The experimental results are shown in *Table 7* as follows. We keep other hyperparameters and experiment configuration consistent. We use GloVe model with 50 dimensions here. Relatively speaking, higher k value may guarantee better results to some extent. It is reasonable because higher k means larger range of neighbourhood that we take into account. To increase the range of neighbourhood, we can better grasp the indirect similarity between two objects.

However, when the k value is too large, we may lose the advantages from the LID, which is suitable for local structures and local characteristics. And also with too large k, the indirect similarity coming from common neighbours become meaningless. Therefore, we should choose an appropriately large value for k in k-Nearest-Neighbours. The best selection for this dataset and configurations is 100.

k in k-NN	Spearman's rho	Pearson's r
25	0.509	0.429
50	0.516	0.441
100	0.522	0.45
200	0.517	0.458

Table 7. The comparisons of results using different k values

4.2. Text Classification with LID

We put forward context-based LID, as a replacement for TF-IDF, to highlight the informative words in a document. We used the whole document as the context to calculate the LID for each word. We then used this context-based LID as the weight to get the document vectors. The experiment results show that the context-based LID is as effective as TF-IDF in finding the most informative words in documents.

4.2.1. Datasets

The 20Newsgroups dataset is one of the benchmarks in document classification. It is a collection of approximately 18000 newsgroup documents, labelled with 20 categories. The average length of the documents in the collection is 492. The 20NewsGroups dataset is popular for experiments in text clustering and text classification. Some of the categories are highly related and they can constitute another higher-level topic according to the subject matter. There are hierarchical categories in the 20Newsgroups collection. The table below shows the different topics in 20Newsgroups.

Classes	Subclasses
comp	comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, comp.windows.x
rec	rec.autos, rec.motorcycles, rec.sport.baseball, rec.sport.hockey
sci	sci.crypt, sci.electronics, sci.med, sci.space
misc	misc.forsale
talk	talk.politics.misc, talk.politics.guns, talk.politics.mideast
religion	talk.religion.misc, alt.atheism, soc.religion.christian

Table 8. Categories details about 20Newsgroup

There are multiple versions of the 20Newsgroup collections and we choose the one embedded in Scikit-learn. The 20Newsgroups collection is divided into 2 subsets, the training set with

11308 documents and test set with 7526 documents. Every document in the collection is with timestamp information indicating the posted time of the news. The partition is based on a specific date. The training sets are posted before that date while the test sets are more recent and posted after that.

4.2.2. Baselines

Regarding the baseline methods, we choose some traditional approaches such as Bag-of-word and its TFIDF variation and the document vectors with IDF as weights.

Bag-of-word: The most 10000 frequent words from the whole training sets are chosen. Because by filtering out some rare words, we can restrict the dimension of the feature vectors, accelerate the training process and avoid overfitting. The counts of the word frequency of occurrence in a document serve as the features for that document sample.

Bag-of-word + TFIDF: Similarly, the most frequent 10000 words are kept in the TF-IDF variation of BoW. Simply, the feature vectors of a document are composed of the TF-IDF values for each feature word regarding that document.

Doc2Vec (IDF): We use the same setting of pre-trained word vectors from fastText in this method. And then the IDF of each word considering the whole collection is assigned as the weight for the word vectors. The dimension of the Doc2Vec is 300, the same as the pre-trained word vectors.

4.2.3. Experiments Setting

Data Pre-processing

In natural language processing, removing morphology of words is an important data pre-processing stage. Inflectional morphology and derivational morphology are the two principal types of morphology in English. Inflectional morphology creates plenty of grammatical variants, mainly for nouns, verbs, and adjectives. Lemmatisation is the process to remove any inflection to obtain the uninflected form, called lemma. In general, Inflectional morphology doesn’t change the word class and the main word senses of the original word, but derivational morphology is different which usually create distinct words. In English, derivational suffixes often change the lexical category while prefixes change the meaning of words without changing the lexical category. Stemming is the process to strip off all suffixes and leave a stem.

Compared to lemmatisation, the result of stemming, the stem are usually not a lexical term. Stemming can provide less lexical sparsity than lemmatisation, and thus it is more popular in information retrieval. In the table below, we explain the two kinds of morphology, lemmatisation and stemming with some representative examples.

	Explanation with examples	
Inflectional morphology	Nouns: number of noun Verbs: number of subject, the aspect and the tense of the action Adjectives: comparatives and superlatives	car -> cars (-s) stop -> stopping, stopped (-ing, -ed) small -> smaller, smallest (-er, -est)
Derivational morphology	Suffixes: change the lexical category Prefixes: change the meaning	rapid -> rapidly (-ly) write -> writer (-er) final -> finalise (-ise) write -> rewrite(re-) happy -> unhappy (un-)
Lemmatization	speaking -> speak, watches -> watch ships -> ship bigger -> big, tallest -> tall	
Stemming	automate, automatic, automation -> automat	

Table 9. Explanation and examples of pre-processing

In the task of text classification, we decide to utilise the lemmatisation to remove inflectional morphology in the documents. After this data pre-processing, only the lemmas remain in the vocabulary dictionary of the corpus. Lemmatization is proven that it may improve the accuracy of some practical information extraction and text mining tasks. Similarly, our purpose is to differentiate the more or less relevant and informative words in the documents by LID. We assume that our algorithm can benefit from the lemmatisation process.

Approximation of Calculation

As the number of the words in the word lists increase, to identify the exact k-Nearest-Neighbours for each word can be very time-consuming with high computational complexity. The exact computation can hardly be efficient with regard to huge size of datasets and high dimension. Therefore, in order to accelerate the computation of LID, we used some toolkits to get the approximation of k-nearest neighbours of the reference word vector. Non-Metric Space

Library (NMSLIB)[37] is an efficient similarity search library and a toolkit for evaluation of k-NN methods for generic non-metric spaces. It focuses on approximate methods for k-NN queries. With the same hardware, it takes half an hour to obtain the exact k-NN for each reference point. But using NMSLIB library, we can finish the calculation of approximate k-NN in less than a minute. The table below compares the running time of two approaches for getting k-NN, and in this experiment, we used the 25kWordLists for testing.

Datasets Details	25k words, 50-dimension	25k words, 300-dimension
Running time with NMSLIB	16.57s	43.86s
Running time of exact k-NN	717.42s	1736.92s

Table 10. Running time comparisons of two k-NN methods

We are also concerned with the accuracy of the approximation. Even though the approximate method can improve efficiency remarkably, we need to validate the accuracy of the approximation. We set k = 100 and then compare the results between exact k-NN and approximate k-NN. We randomly selected 1000 words from the word lists and applied these two methods to the calculation of k-NN. We count the number of the different words between two k-NN results and took the average of all samples. The mean error rate is 12.9%, which is acceptable. We need to consider the trade-off between accuracy and efficiency. Because the approximate method can solve the problems much faster with a low error rate, we decide to use the approximate method for all the downstream tasks.

Configuration

In order to avoid imbalanced classes, we mainly utilize high-level classes in the 20Newsgroups, including 6 classes. To simplify the task, we use pre-trained word vectors from some word embedding models rather than train the word vectors by ourselves. By default, we implement the system with 300-dimension word vectors from fastText model. After obtaining the document vector, which has the same dimension as its components, the word vector, we feed the classifiers with the document vectors as the features. Most of the classic machine learning-based classifiers are applied to test the effectiveness of document vectors with context-based LID. They include Support Vector Machine classifier, Logistic Regression classifier, Random Forest classifier, k-Nearest-Neighbours classifier and Artificial Neural Network(Multi-layer Perceptron) classifier. We used cross-validation for some classifiers to figure out the most appropriate hyper-parameters.

4.2.4. Results and Analysis

The evaluation metric of the dataset is the Macro-F1 measure. The table below illustrates the performance of our approach and some baselines.

Methods(Features)	Classifiers				
	LR	SVM	Random Forest	k-NN	ANN
BoW	0.82	0.72	0.81	0.80	0.82
BoW (TF-IDF)	0.88	0.88	0.83	0.82	**0.90**
Doc2Vec (IDF)	0.84	0.84	0.79	0.81	0.87
Doc2Vec (LID) k = 100	0.85	0.86	0.82	0.81	**0.89**
Doc2Vec (LID) k = 50	0.84	0.85	0.79	0.80	0.87
Doc2Vec (LID) k = 20	0.82	0.82	0.78	0.77	0.83

Table 11. F1-score of text classification

When we compare the results of the traditional BoW approaches and the distributed representation of documents approaches, the BoW methods outperform the document vectors methods with most of the classifiers. But we can see that the document vectors with LID as weights($k = 100$) achieve the second highest F1 score with slight difference from BoW-TFIDF.

With regard to the efficiency of the systems, for Doc2Vec, the running time of the training process of all classifiers is much lower than that of BoW methods. For instance, for the 20Newsgroups, most classifiers can be trained in less than 30 seconds with Doc2Vec features. In this sense, the low-dimensional document vectors are more efficient in training.

Besides, apparently, the dimension of the feature vectors in BoW is much higher than that of distributed representation of documents. Because of the dimension reduction, the most relevant and informative features are preserved. It can avoid overfitting naturally. Comparatively, the BoW feature vectors usually suffer from data sparsity and high dimensionality, while these problems can be easily tackled when using low-dimensional Doc2Vec. In addition, besides using the Doc2Vec for text classification, the low-dimensional distributed representation of text is practical and appropriate for some downstream NLP tasks.

With respect to the comparison of Doc2Vec with IDF and Doc2Vec with context-based LID, these two methods achieve similar performance through different classifiers. We can claim that the context-based LID performs well in capturing the contextual information as TF-IDF does.

4.2.5. Hyperparameters and Important Factors

Context and k

The size of the context to be taken into account is of great significance. We tried a different range of context to calculate the LID of words in a document, including a paragraph, a document, a subset of documents, the full collection of documents. When we take other documents in the collection into consideration, we attempt to capture not only the interactions of words in the target document, but also the relationship between the centred word and the collection of documents. *Figure 7* shows the change of F1 score using different types of context using the same classifier and experiment configurations. For the document classification in 20Newsgroups, a document is the best choice of context. We infer that it is because the unit in the collection is a document and each document is relatively independent that sets of multiple documents cannot make more sense. Comparatively, the average length of the paragraphs in the corpus is too short and thus the paragraph context cannot offer plentiful useful global information.

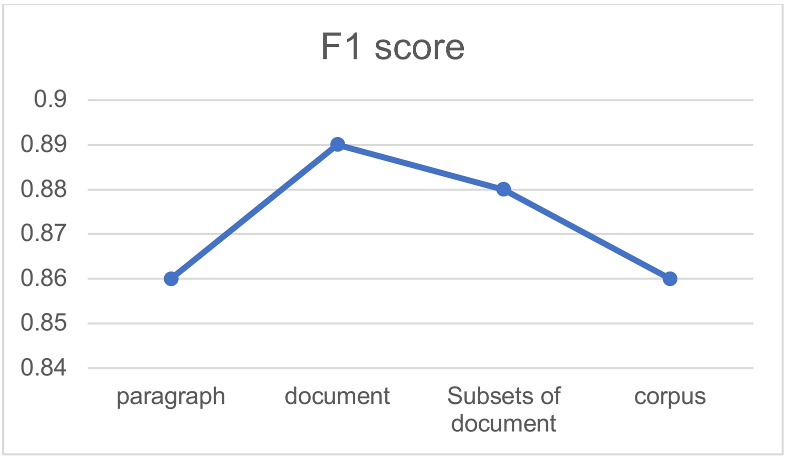


Figure 7.F1-score comparisons of different contexts

Besides, the value of k in k-Nearest-Neighbours is another important hyper-parameter. In general, the selection of k relies on the range of context we are using. For example, if we take the whole document as the context, because the average length of documents is around 500, we attempt to use 20, 50 and 100 as the value of k. From the experiment results above, we can observe that k = 100 is the best selection for the highest F1 score. Generally speaking, relatively higher k value may guarantee better results to some extent.

Pre-trained Word Vectors and Dimensions

As we discussed in the investigation of word LID, the pre-trained word vectors play a fundamental role in calculating LID. In fact, in Doc2Vec with LID, the pre-trained word vectors serve as the foundational bricks in our system. We tried different word embedding models and selected various dimension of word vectors, to test the influence of the word embeddings on Doc2Vec. Word2Vec, GloVe and fastText are the three word embedding methods that we used.

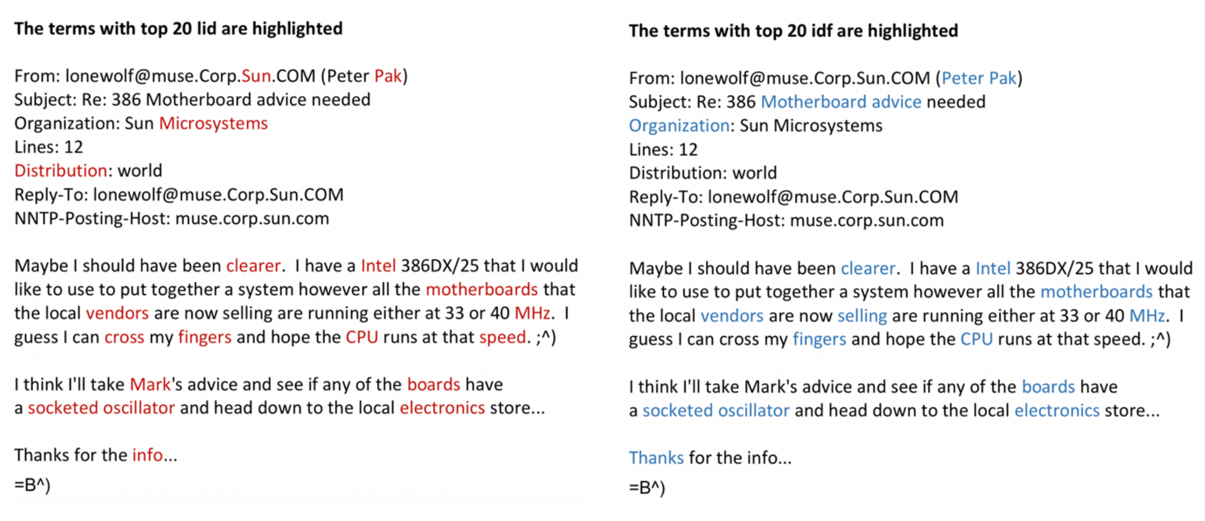
The dimension of word vectors is another hyperparameter. And we suppose that the higher dimension of word vectors we use, the more semantic information that is kept and better presentation of LID. With regard to the dimensions of the word vectors, we tried 50, 100, 200 and 300 with GloVe model. Generally speaking, based on the experiment results, the higher dimension of the word vectors and Doc2Vec, the more accurate prediction we can make. By appropriately increasing the dimension of the distributed representations, more semantic and contextual information is captured. This information is the key to text classification. *Table 12* below illustrates the comparisons of results with different pre-trained vectors or dimensions.

Pre-trained vectors	Dimensions	F1-score
fastText	300	0.89
GloVe	50	0.83
	100	0.85
	200	0.85
	300	0.86
Word2Vec	300	0.83

Table 12. F1-score comparisons with different word vectors and dimensions

4.2.6. Visualisation of LID Effectiveness

In the example below, we selected a document from the category "computer" and we highlighted the terms with the top 20 highest IDF and context-based LID respectively. Obviously, these two sets of words have many words in common. Also based on human comprehension of the topic "computer", it is obvious that the highlighted words are highly informative and related to the topic. We can infer that the context-based LID can indicate the more relevant and informative words in the text, similar to TF-IDF.



The terms with top 20 lid are highlighted

From: lonewolf@muse.Corp.Sun.COM (Peter Pak)
Subject: Re: 386 Motherboard advice needed
Organization: Sun Microsystems
Lines: 12
Distribution: world
Reply-To: lonewolf@muse.Corp.Sun.COM
NNTP-Posting-Host: muse.corp.sun.com

Maybe I should have been clearer. I have a Intel 386DX/25 that I would like to use to put together a system however all the motherboards that the local vendors are now selling are running either at 33 or 40 MHz. I guess I can cross my fingers and hope the CPU runs at that speed. ;^)

I think I'll take Mark's advice and see if any of the boards have a socketed oscillator and head down to the local electronics store...

Thanks for the info...

=B^)

The terms with top 20 idf are highlighted

From: lonewolf@muse.Corp.Sun.COM (Peter Pak)
Subject: Re: 386 Motherboard advice needed
Organization: Sun Microsystems
Lines: 12
Distribution: world
Reply-To: lonewolf@muse.Corp.Sun.COM
NNTP-Posting-Host: muse.corp.sun.com

Maybe I should have been clearer. I have a Intel 386DX/25 that I would like to use to put together a system however all the motherboards that the local vendors are now selling are running either at 33 or 40 MHz. I guess I can cross my fingers and hope the CPU runs at that speed. ;^)

I think I'll take Mark's advice and see if any of the boards have a socketed oscillator and head down to the local electronics store...

Thanks for the info...

=B^)

Figure 8. Examples of informative words with highest LID OR IDF

We also did some other trails to validate the effectiveness of LID and the underlying attention mechanism. We are concerned with the document LID according to different categories in this experiment. We used the weighted average of all the word vectors to get the basic representation of document vectors. For the text classification task, we then compute the LID of document based on different categories, e.g. science, fiction and press (the categories of the documents in the corpus). We used bag-of-words features as a baseline and we found that adding category-based LID features could slightly improve the performance. We measured the importance of the features as well. For example, with the Random Forest Classifier, the most important and informative features can be selected according to either the information gain or Gini impurity. We can observe that the category-based LID of document is highly ranked.

5. Conclusion and Future Work

5.1. Conclusion and Contributions

In conclusion, based on the investigation of word LID with some generic word lists, LID can represent some interesting lexical and semantic information of words. We see some relationships between LID and the attributes of words, like word frequency, lexical category and word meaning. LID is also perfectly suitable for similarity-based tasks as well, such as word similarity.

Furthermore, we then employ LID in long texts and put forward a new model, context-based LID. Utilizing context-based LID as the weighting schema, we can observe relatively high accuracy in the experiments of document classification. We can claim that the context-based LID performs well in capturing the contextual information, similar to TFIDF. It can be of value in some information retrieval tasks, such as keyword extraction.

Last but not least, our method provides some new insights into the applications of LID and the distributed representation in natural language processing. We further prove the generic capability and effectiveness of LID in some classification tasks and similarity search tasks.

Our contributions are of three-folds:

Firstly, we investigate the LID in a novel field of intrinsic dimensionality, textual data, which notably differs from common structural data. We apply the LID in natural language processing and find some interesting stories about word LID.

Secondly, we propose a new approach to text classification encoding the LID information of words. In comparison to document vectors with IDF weighting, our method outperforms Doc2Vec with IDF in all classifiers. The results demonstrate that our method can represent the document with low-dimension vectors and maintain most of the significant information as well.

Thirdly, we introduce the context-based LID, which is proven to be a good sign of word relevancy. It also provides insights into which words in the document are of more importance during the classification. With some visualization tools, context-based LID could be valuable in the interpretation of machine learning and information retrieval systems.

5.2. Future Work

Most of the word embedding methods claim that they can capture the semantic and syntactic information of words. To some extent, the meaning of the word is encoded in the low-dimensional vector. In WordNet, the meaning or senses of words are described as a set of synsets. It is interesting to investigate and make comparisons of the LIDs among the words sharing the same synset.

We utilized the context-based LID in document classification. We plan to apply the context-based LID in other text classification applications, such as sentiment analysis. Meanwhile, we intend to employ the context-based LID model on short pieces of texts. For example, different from documents, sentence or paragraph is shorter textual data. These short texts are ubiquitous, such as tweets, text messages, and news headlines. Short texts usually suffer from the sparsity and noise in their use of vocabulary. To represent the short text, traditional bag-of-word feature vector seems to be tedious and sparse. Distributional representation of text may be a better option.

Inspired by the common properties and similarity of TF-IDF and LID, we plan to investigate the use of LID in keyword extraction. Keyword extraction is one of the important problems in Information Retrieval, Text Mining and Natural Language Processing. Keywords refer to the words that best summarise the subject of the document. In general, the terms which are more related to the topic or category of the document can better describe the document overall. We assume LID can be beneficial to the automatic identification of keywords.

Reference

[1] Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1988. Learning representations by back-propagating errors. Cognitive modeling, 5(3), p.1.

[2] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. and Harshman, R., 1990. Indexing by latent semantic analysis. Journal of the American society for information science, 41(6), pp.391-407.

[3] Blei, D.M., Ng, A.Y. and Jordan, M.I., 2003. Latent dirichlet allocation. Journal of machine Learning research, 3(Jan), pp.993-1022.

[4] Schneider, K.M., 2005, October. Weighted average pointwise mutual information for feature selection in text categorization. In European Conference on Principles of Data Mining and Knowledge Discovery (pp. 252-263). Springer, Berlin, Heidelberg.

[5] Evert, S., 2005. The statistics of word cooccurrences (Doctoral dissertation, Dissertation, Stuttgart University).

[6] Golub, G.H. and Van Loan, C.F., 2012. Matrix computations (Vol. 3). JHU press.

[7] Lee, D.D. and Seung, H.S., 2001. Algorithms for non-negative matrix factorization. In Advances in neural information processing systems (pp. 556-562).

[8] Mikolov, T., Chen, K., Corrado, G. and Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[9] Pennington, J., Socher, R. and Manning, C., 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

[10] Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T., 2017. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, pp.135-146.

[11] Baroni, M., Dinu, G. and Kruszewski, G., 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Vol. 1, pp. 238-247).

[12] Karger, D.R. and Ruhl, M., 2002, May. Finding nearest neighbors in growth-restricted metrics. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*(pp. 741-750). ACM.

[13] Houle, M.E., 2013, December. Dimensionality, discriminability, density and distance distributions. In *2013 IEEE 13th International Conference on Data Mining Workshops* (pp. 468-473). IEEE.

[14] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. and Harshman, R., 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, *41*(6), pp.391-407.

[15] Hofmann, T., 1999, July. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence* (pp. 289-296). Morgan Kaufmann Publishers Inc..

[16] De Boom, C., Van Canneyt, S., Demeester, T. and Dhoedt, B., 2016. Representation learning for very short texts using weighted word embedding aggregation. *Pattern Recognition Letters*, *80*, pp.150-156.

[17] Chen, M., 2017. Efficient vector representation for documents through corruption. *arXiv preprint arXiv:1707.02377*.

[18] Conneau, A., Schwenk, H., Barrault, L. and Lecun, Y., 2016. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781*.

[19] Lai, S., Xu, L., Liu, K. and Zhao, J., 2015, February. Recurrent convolutional neural networks for text classification. In Twenty-ninth AAAI conference on artificial intelligence.

[20] Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T., 2016. Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759.

[21] Le, Q. and Mikolov, T., 2014, January. Distributed representations of sentences and documents. In International conference on machine learning (pp. 1188-1196).

[22] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A. and Hovy, E., 2016. Hierarchical attention networks for document classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 1480-1489).

[23] Bouveyron, C., Celeux, G. and Girard, S., 2011. Intrinsic dimension estimation by maximum likelihood in isotropic probabilistic PCA. Pattern Recognition Letters, 32(14), pp.1706-1713.

[24] Camastra, F. and Vinciarelli, A., 2002. Estimating the intrinsic dimension of data with a fractal-based method. IEEE Transactions on pattern analysis and machine intelligence, 24(10), pp.1404-1407.

[25] Costa, J.A. and Hero, A.O., 2003, November. Entropic graphs for manifold learning. In The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003 (Vol. 1, pp. 316-320). IEEE.

[26] Bruske, J. and Sommer, G., 1998. Intrinsic dimensionality estimation with optimally topology preserving maps. IEEE Transactions on pattern analysis and machine intelligence, 20(5), pp.572-575.

[27] Houle, M.E., Kashima, H. and Nett, M., 2012, December. Generalized expansion dimension. In 2012 IEEE 12th International Conference on Data Mining Workshops (pp. 587-594). IEEE.

[28] Rozza, A., Lombardi, G., Ceruti, C., Casiraghi, E. and Campadelli, P., 2012. Novel high intrinsic dimensionality estimators. Machine learning, 89(1-2), pp.37-65.

[29] Amsaleg, L., Chelly, O., Furon, T., Girard, S., Houle, M.E., Kawarabayashi, K.I. and Nett, M., 2015, August. Estimating local intrinsic dimensionality. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 29-38). ACM.

[30] Casanova, G., Englmeier, E., Houle, M.E., Kröger, P., Nett, M., Schubert, E. and Zimek, A., 2017. Dimensional testing for reverse k-nearest neighbor search. Proceedings of the VLDB Endowment, 10(7), pp.769-780.

[31] Romano, S., Chelly, O., Nguyen, V., Bailey, J. and Houle, M.E., 2016, December. Measuring dependency via intrinsic dimensionality. In 2016 23rd International Conference on Pattern Recognition (ICPR) (pp. 1207-1212). IEEE.

[32] Von Brünken, J., Houle, M.E. and Zimek, A., 2015. Intrinsic dimensional outlier detection in high-dimensional data. Technical report, National Institute of Informatics, Tokyo.

[33] Ma, X., Li, B., Wang, Y., Erfani, S.M., Wijewickrema, S., Schoenebeck, G., Song, D., Houle, M.E. and Bailey, J., 2018. Characterizing adversarial subspaces using local intrinsic dimensionality. arXiv preprint arXiv:1801.02613.

[34] Amsaleg, L., Bailey, J., Barbe, D., Erfani, S., Houle, M.E., Nguyen, V. and Radovanović, M., 2017, December. The vulnerability of learning to adversarial perturbation increases with intrinsic dimensionality. In 2017 IEEE Workshop on Information Forensics and Security (WIFS) (pp. 1-6). IEEE.

[35] Houle, M.E., Oria, V., Rohloff, K.R. and Wali, A.M., 2018, October. LID-Fingerprint: A Local Intrinsic Dimensionality-Based Fingerprinting Method. In International Conference on Similarity Search and Applications (pp. 134-147). Springer, Cham.

[36] Ma, X., Wang, Y., Houle, M.E., Zhou, S., Erfani, S.M., Xia, S.T., Wijewickrema, S. and Bailey, J., 2018. Dimensionality-driven learning with noisy labels. arXiv preprint arXiv:1806.02612.

[37] Malkov, Y.A. and Yashunin, D.A., 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE transactions on pattern analysis and machine intelligence.

[38] Hassan, S.H. and Mihalcea, R., 2011, August. Semantic relatedness using salient semantic analysis. In Twenty-Fifth AAAI Conference on Artificial Intelligence.

[39] Speer, R., Chin, J. and Havasi, C., 2017, February. Conceptnet 5.5: An open multilingual graph of general knowledge. In Thirty-First AAAI Conference on Artificial Intelligence.

[40] Hassan, S., Banea, C. and Mihalcea, R., 2012, June. Measuring semantic relatedness using multilingual representations. In Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (pp. 20-29). Association for Computational Linguistics.

[41] Gabrilovich, E. and Markovitch, S., 2007, January. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In IJcAI (Vol. 7, pp. 1606-1611).