Efficient Estimation of Word Representations in Vector Space

向量空間中單詞表示的有效估計

Tomas Mikolov Google Inc., Mountain View, CA [tmikolov@google.com](mailto:tmikolov@google.com)

Kai Chen Google Inc., Mountain View, CA [kaichen@google.com](mailto:kaichen@google.com)

Greg Corrado Google Inc., Mountain View, CA [gcorrado@google.com](mailto:gcorrado@google.com)

Jeffrey Dean Google Inc., Mountain View, CA [jeff@google.com](mailto:jeff@google.com)

Tomas Mikolov Google Inc.，加利福尼亞山景城tmikolov@google.com

Kai Chen Google Inc.，加利福尼亞山景城kaichen@google.com

Greg Corrado Google Inc.，加利福尼亞山景城gcorrado@google.com

Jeffrey Dean Google Inc.，加利福尼亞州山景城jeff@google.com

Abstract

摘要

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

我們提出了兩種新穎的模型體系結構，用於從非常大的數據集中計算單詞的連續矢量表示。 在單詞相似性任務中測量這些表示的質量，並將結果與基於不同類型的神經網絡的以前性能最佳的技術進行比較。 我們觀察到準確性的大幅提高，而計算成本卻低得多，即從16億個單詞的數據集中學習高質量的單詞向量只需不到一天的時間。 此外，我們證明了這些向量在我們的測試集上提供了最新的性能，用於測量句法和語義詞的相似性。

1 Introduction

1引言

Many current NLP systems and techniques treat words as atomic units - there is no notion of similarity between words, as these are represented as indices in a vocabulary. This choice has several good reasons - simplicity, robustness and the observation that simple models trained on huge amounts of data outperform complex systems trained on less data. An example is the popular N-gram model used for statistical language modeling - today, it is possible to train N-grams on virtually all available data (trillions of words [3]).

當前的許多NLP系統和技術都將單詞視為原子單位-單詞之間沒有相似性的概念，因為它們在詞彙表中表示為索引。 這種選擇有幾個很好的理由-簡單性，魯棒性和觀察到，對大量數據進行訓練的簡單模型優於對較少數據進行訓練的複雜系統。 一個例子是用於統計語言建模的流行N-gram模型-如今，可以在幾乎所有可用數據（萬億個單詞[3]）上訓練N-gram。

However, the simple techniques are at their limits in many tasks. For example, the amount of relevant in-domain data for automatic speech recognition is limited - the performance is usually dominated by the size of high quality transcribed speech data (often just millions of words). In machine translation, the existing corpora for many languages contain only a few billions of words or less. Thus, there are situations where simple scaling up of the basic techniques will not result in any significant progress, and we have to focus on more advanced techniques.

但是，簡單的技術在許多任務中都處於極限。 例如，用於自動語音識別的相關域內數據量有限-性能通常由高質量的轉錄語音數據（通常只有數百萬個單詞）的大小決定。 在機器翻譯中，許多語言的現有語料庫僅包含數十億個單詞或更少。 因此，在某些情況下，簡單地擴展基本技術不會導致任何重大進展，我們必須專注於更高級的技術。

With progress of machine learning techniques in recent years, it has become possible to train more complex models on much larger data set, and they typically outperform the simple models. Probably the most successful concept is to use distributed representations of words [10]. For example, neural network based language models significantly outperform N-gram models [1, 27, 17].

近年來，隨著機器學習技術的進步，已經可以在更大的數據集上訓練更複雜的模型，並且它們通常勝過簡單的模型。 可能最成功的概念是使用單詞的分佈式表示[10]。 例如，基於神經網絡的語言模型明顯優於N-gram模型[1、27、17]。

1.1 Goals of the Paper

1.1論文目標

The main goal of this paper is to introduce techniques that can be used for learning high-quality word vectors from huge data sets with billions of words, and with millions of words in the vocabulary. As far as we know, none of the previously proposed architectures has been successfully trained on more than a few hundred of millions of words, with a modest dimensionality of the word vectors between 50 - 100.

本文的主要目的是介紹可用於從海量數據集和數十億個單詞的海量數據集中學習高質量單詞向量的技術。 據我們所知，沒有一個先前提出的架構已經成功地針對超過幾億個單詞進行了訓練，單詞向量的維數在50至100之間。

We use recently proposed techniques for measuring the quality of the resulting vector representations, with the expectation that not only will similar words tend to be close to each other, but that words can have multiple degrees of similarity [20]. This has been observed earlier in the context of inflectional languages - for example, nouns can have multiple word endings, and if we search for similar words in a subspace of the original vector space, it is possible to find words that have similar endings [13, 14].

我們使用最近提出的技術來測量所得矢量表示的質量，並期望不僅相似的單詞趨於彼此接近，而且單詞可以具有多個相似度[20]。 這在較早的屈折語言中就已觀察到-例如，名詞可以有多個詞尾，如果我們在原始向量空間的子空間中搜索相似的詞，就有可能找到具有相似的詞尾[13 ，14]。

Somewhat surprisingly, it was found that similarity of word representations goes beyond simple syntactic regularities. Using a word offset technique where simple algebraic operations are performed on the word vectors, it was shown for example that vector(”King”) - vector(”Man”) + vector(”Woman”) results in a vector that is closest to the vector representation of the word Queen [20].

令人驚訝地發現，單詞表示的相似性超出了簡單的句法規律性。 使用對詞向量執行簡單代數運算的詞偏移技術，例如，顯示vector（“ King”）-vector（“ Man”）+ vector（“ Woman”）產生的向量最接近 皇后[20]一詞的向量表示。

In this paper, we try to maximize accuracy of these vector operations by developing new model architectures that preserve the linear regularities among words. We design a new comprehensive test set for measuring both syntactic and semantic regularities1 , and show that many such regularities can be learned with high accuracy. Moreover, we discuss how training time and accuracy depends on the dimensionality of the word vectors and on the amount of the training data.

在本文中，我們試圖通過開發保留單詞之間線性規則性的新模型架構來最大程度地提高這些向量運算的準確性。 我們設計了一種新的綜合測試集，用於測量句法和語義規律性1，並表明可以高精度地學習許多這樣的規律性。 此外，我們討論了訓練時間和準確性如何取決於單詞向量的維數和訓練數據的數量。

1.2 Previous Work

1.2前置作業

Representation of words as continuous vectors has a long history [10, 26, 8]. A very popular model architecture for estimating neural network language model (NNLM) was proposed in [1], where a feedforward neural network with a linear projection layer and a non-linear hidden layer was used to learn jointly the word vector representation and a statistical language model. This work has been followed by many others.

單詞作為連續向量的表示已有很長的歷史[10，26，8]。 在[1]中提出了一種非常流行的用於估計神經網絡語言模型（NNLM）的模型架構，其中使用具有線性投影層和非線性隱藏層的前饋神經網絡來共同學習單詞矢量表示和統計 語言模型。 這項工作已被其他許多人關注。

Another interesting architecture of NNLM was presented in [13, 14], where the word vectors are first learned using neural network with a single hidden layer. The word vectors are then used to train the NNLM. Thus, the word vectors are learned even without constructing the full NNLM. In this work, we directly extend this architecture, and focus just on the first step where the word vectors are learned using a simple model.

NNLM的另一種有趣的架構在[13，14]中提出，其中首先使用具有單個隱藏層的神經網絡來學習單詞向量。 然後，將詞向量用於訓練NNLM。 因此，即使不構建完整的NNLM，也可以學習單詞向量。 在這項工作中，我們直接擴展此體系結構，並只專注於使用簡單模型學習單詞向量的第一步。

It was later shown that the word vectors can be used to significantly improve and simplify many NLP applications [4, 5, 29]. Estimation of the word vectors itself was performed using different model architectures and trained on various corpora [4, 29, 23, 19, 9], and some of the resulting word vectors were made available for future research and comparison2 . However, as far as we know, these architectures were significantly more computationally expensive for training than the one proposed in [13], with the exception of certain version of log-bilinear model where diagonal weight matrices are used [23].

後來顯示，單詞向量可用於顯著改善和簡化許多NLP應用程序[4、5、29]。 單詞向量本身的估計是使用不同的模型架構進行的，並在各種語料庫上進行了訓練[4、29、23、19、9]，並且一些所得的單詞向量可供將來的研究和比較使用2。 然而，據我們所知，這些體系結構的訓練在計算上比[13]中提出的體系結構要昂貴得多，除了某些版本的對數雙線性模型使用對角線權重矩陣[23]之外。

2 Model Architectures

2模型架構

Many different types of models were proposed for estimating continuous representations of words, including the well-known Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). In this paper, we focus on distributed representations of words learned by neural networks, as it was previously shown that they perform significantly better than LSA for preserving linear regularities among words [20, 31]; LDA moreover becomes computationally very expensive on large data sets.

提出了許多不同類型的模型來估計單詞的連續表示，包括著名的潛在語義分析（LSA）和潛在狄利克雷分配（LDA）。 在本文中，我們專注於由神經網絡學習的單詞的分佈式表示，因為先前證明它們在保持單詞之間的線性規則性方面比LSA表現更好[20，31]。 此外，在大型數據集上，LDA在計算上變得非常昂貴。

Similar to [18], to compare different model architectures we define first the computational complexity of a model as the number of parameters that need to be accessed to fully train the model. Next, we will try to maximize the accuracy, while minimizing the computational complexity.

類似於[18]，為了比較不同的模型架構，我們首先將模型的計算複雜度定義為完全訓練模型所需的參數數量。 接下來，我們將嘗試使準確性最大化，同時使計算複雜度最小化。

For all the following models, the training complexity is proportional to

O = E × T × Q, (1)

where E is number of the training epochs, T is the number of the words in the training set and Q is defined further for each model architecture. Common choice is E = 3 − 50 and T up to one billion. All models are trained using stochastic gradient descent and backpropagation [26].

對於以下所有模型，訓練複雜度與

O = E×T×Q（1）

其中E是訓練紀元的數量，T是訓練集中的單詞數，並且針對每種模型架構進一步定義Q。 常見的選擇是E = 3 − 50和T高達十億。 所有模型都使用隨機梯度下降和反向傳播訓練[26]。

2.1 Feedforward Neural Net Language Model (NNLM)

2.1前饋神經網絡語言模型（NNLM）

The probabilistic feedforward neural network language model has been proposed in [1]. It consists of input, projection, hidden and output layers. At the input layer, N previous words are encoded using 1-of-V coding, where V is size of the vocabulary. The input layer is then projected to a projection layer P that has dimensionality N × D, using a shared projection matrix. As only N inputs are active at any given time, composition of the projection layer is a relatively cheap operation.

文獻[1]中提出了概率前饋神經網絡語言模型。 它由輸入，投影，隱藏和輸出層組成。 在輸入層，使用V的1編碼對N個先前的單詞進行編碼，其中V是詞彙量。 然後，使用共享的投影矩陣將輸入層投影到尺寸為N×D的投影層P上。 由於在任何給定時間只有N個輸入處於活動狀態，因此投影層的組成是相對便宜的操作。

The NNLM architecture becomes complex for computation between the projection and the hidden layer, as values in the projection layer are dense. For a common choice of N = 10, the size of the projection layer (P) might be 500 to 2000, while the hidden layer size H is typically 500 to 1000 units. Moreover, the hidden layer is used to compute probability distribution over all the words in the vocabulary, resulting in an output layer with dimensionality V . Thus, the computational complexity per each training example is

Q = N × D + N × D × H + H × V, (2)

where the dominating term is H × V . However, several practical solutions were proposed for avoiding it; either using hierarchical versions of the softmax [25, 23, 18], or avoiding normalized models completely by using models that are not normalized during training [4, 9]. With binary tree representations of the vocabulary, the number of output units that need to be evaluated can go down to around log2(V ). Thus, most of the complexity is caused by the term N × D × H.

NNLM體系結構在投影層和隱藏層之間的計算變得很複雜，因為投影層中的值很密集。對於N = 10的常見選擇，投影層（P）的大小可能為500到2000，而隱藏層的大小H通常為500到1000個單位。此外，隱藏層用於計算詞彙表中所有單詞的概率分佈，從而產生維度為V的輸出層。因此，每個訓練示例的計算複雜度為

Q = N×D + N×D×H + H×V，（2）

其中主導項是H×V。但是，為避免這種情況提出了一些實際的解決方案。要么使用softmax的分層版本[25，23，18]，要么使用在訓練過程中未規範化的模型來完全避免歸一化模型[4，9]。用詞彙表的二叉樹表示，需要評估的輸出單位數可以降低到log2（V）左右。因此，大多數複雜性是由項N×D×H引起的。

In our models, we use hierarchical softmax where the vocabulary is represented as a Huffman binary tree. This follows previous observations that the frequency of words works well for obtaining classes in neural net language models [16]. Huffman trees assign short binary codes to frequent words, and this further reduces the number of output units that need to be evaluated: while balanced binary tree would require log2(V ) outputs to be evaluated, the Huffman tree based hierarchical softmax requires only about log2(Unigram perplexity(V )). For example when the vocabulary size is one million words, this results in about two times speedup in evaluation. While this is not crucial speedup for neural network LMs as the computational bottleneck is in the N ×D×H term, we will later propose architectures that do not have hidden layers and thus depend heavily on the efficiency of the softmax normalization.

在我們的模型中，我們使用層次化softmax，其中詞彙表被表示為霍夫曼二叉樹。 這是根據先前的觀察結果得出的，即詞頻可以很好地在神經網絡語言模型中獲得類[16]。 霍夫曼樹將短的二進制代碼分配給頻繁的單詞，這進一步減少了需要評估的輸出單元的數量：雖然平衡的二叉樹將需要評估log2（V）輸出，但是基於霍夫曼樹的分層softmax僅需要約log2 （Unigram困惑度（V））。 例如，當詞彙量為一百萬個單詞時，這會使評估速度提高大約兩倍。 雖然這對於神經網絡LM而言不是關鍵的提速，因為計算瓶頸在N×D×H術語中，但我們稍後將提出不具有隱藏層的架構，因此在很大程度上取決於softmax歸一化的效率。

2.2 Recurrent Neural Net Language Model (RNNLM)

2.2遞歸神經網絡語言模型（RNNLM）

Recurrent neural network based language model has been proposed to overcome certain limitations of the feedforward NNLM, such as the need to specify the context length (the order of the model N), and because theoretically RNNs can efficiently represent more complex patterns than the shallow neural networks [15, 2]. The RNN model does not have a projection layer; only input, hidden and output layer. What is special for this type of model is the recurrent matrix that connects hidden layer to itself, using time-delayed connections. This allows the recurrent model to form some kind of short term memory, as information from the past can be represented by the hidden layer state that gets updated based on the current input and the state of the hidden layer in the previous time step.

已經提出了基於遞歸神經網絡的語言模型來克服前饋NNLM的某些限制，例如需要指定上下文長度（模型N的順序），並且因為理論上RNN可以比淺層神經有效地表示更複雜的模式 網絡[15，2]。 RNN模型沒有投影層。 僅輸入，隱藏和輸出層。 這種類型的模型的特殊之處在於遞歸矩陣，它使用延時連接將隱藏層與其自身相連。 這允許循環模型形成某種短期記憶，因為過去的信息可以由隱層狀態表示，該隱層狀態根據當前輸入和上一時間步的隱層狀態進行更新。

The complexity per training example of the RNN model is

Q = H × H + H × V, (3)

where the word representations D have the same dimensionality as the hidden layer H. Again, the term H × V can be efficiently reduced to H × log2(V ) by using hierarchical softmax. Most of the complexity then comes from H × H.

RNN模型的每個訓練示例的複雜度為

Q = H×H + H×V，（3）

其中，單詞表示D的維數與隱藏層H相同。再次，可以使用分層softmax將術語H×V有效地簡化為H×log2（V）。 然後，大多數複雜度來自H×H。

2.3 Parallel Training of Neural Networks

2.3並行訓練神經網絡

To train models on huge data sets, we have implemented several models on top of a large-scale distributed framework called DistBelief [6], including the feedforward NNLM and the new models proposed in this paper. The framework allows us to run multiple replicas of the same model in parallel, and each replica synchronizes its gradient updates through a centralized server that keeps all the parameters. For this parallel training, we use mini-batch asynchronous gradient descent with an adaptive learning rate procedure called Adagrad [7]. Under this framework, it is common to use one hundred or more model replicas, each using many CPU cores at different machines in a data center.

為了在海量數據集上訓練模型，我們在稱為DistBelief [6]的大規模分佈式框架的基礎上實現了多個模型，包括前饋NNLM和本文提出的新模型。 該框架允許我們並行運行同一模型的多個副本，並且每個副本都通過保留所有參數的集中式服務器同步其梯度更新。 對於這種並行訓練，我們使用稱為Adagrad [7]的自適應學習速率程序進行小批量異步梯度下降。 在此框架下，通常使用一百個或多個模型副本，每個副本在數據中心的不同計算機上使用許多CPU內核。

3 New Log-linear Models

3個新的對數線性模型

In this section, we propose two new model architectures for learning distributed representations of words that try to minimize computational complexity. The main observation from the previous section was that most of the complexity is caused by the non-linear hidden layer in the model. While this is what makes neural networks so attractive, we decided to explore simpler models that might not be able to represent the data as precisely as neural networks, but can possibly be trained on much more data efficiently.

在本節中，我們提出了兩種新的模型體系結構，用於學習單詞的分佈式表示形式，以盡量減少計算複雜性。 上一節的主要觀察結果是，大多數複雜性是由模型中的非線性隱藏層引起的。 雖然這就是使神經網絡如此吸引人的原因，但我們決定探索更簡單的模型，這些模型可能無法像神經網絡那樣精確地表示數據，但可能可以有效地訓練更多的數據。

The new architectures directly follow those proposed in our earlier work [13, 14], where it was found that neural network language model can be successfully trained in two steps: first, continuous word vectors are learned using simple model, and then the N-gram NNLM is trained on top of these distributed representations of words. While there has been later substantial amount of work that focuses on learning word vectors, we consider the approach proposed in [13] to be the simplest one. Note that related models have been proposed also much earlier [26, 8].

新的架構直接遵循我們之前的工作[13，14]中提出的架構，發現神經網絡語言模型可以通過兩個步驟成功地進行訓練：首先，使用簡單模型學習連續的單詞向量，然後使用N- gram NNLM在這些單詞的分佈式表示之上訓練。 雖然後來有大量工作專注於學習單詞向量，但我們認為[13]中提出的方法是最簡單的方法。 請注意，相關模型也已經提早提出[26，8]。

3.1 Continuous Bag-of-Words Model

3.1連續詞袋模型

The first proposed architecture is similar to the feedforward NNLM, where the non-linear hidden layer is removed and the projection layer is shared for all words (not just the projection matrix); thus, all words get projected into the same position (their vectors are averaged). We call this architecture a bag-of-words model as the order of words in the history does not influence the projection. Furthermore, we also use words from the future; we have obtained the best performance on the task introduced in the next section by building a log-linear classifier with four future and four history words at the input, where the training criterion is to correctly classify the current (middle) word. Training complexity is then

Q = N × D + D × log2(V ). (4)

We denote this model further as CBOW, as unlike standard bag-of-words model, it uses continuous distributed representation of the context. The model architecture is shown at Figure 1. Note that the weight matrix between the input and the projection layer is shared for all word positions in the same way as in the NNLM.

首先提出的架構與前饋NNLM相似，其中刪除了非線性隱藏層，並且為所有單詞共享了投影層（不僅是投影矩陣）；因此，所有單詞都投影到同一位置（對它們的向量進行平均）。我們將此架構稱為“詞袋模型”，因為歷史中的詞序不會影響預測。此外，我們也使用未來的話語。通過構建一個對數線性分類器，輸入中包含四個未來和四個歷史單詞，我們的訓練標準是正確分類當前（中間）單詞，從而在下一節介紹的任務上獲得了最佳性能。那麼訓練的複雜性就到了

Q = N×D + D×log2（V）。 （4）

我們將這種模型進一步稱為CBOW，與標準的詞袋模型不同，它使用上下文的連續分佈式表示形式。該模型的體系結構如圖1所示。請注意，輸入和投影層之間的權重矩陣以與NNLM中相同的方式共享給所有單詞位置。

3.2 Continuous Skip-gram Model

3.2連續跳過圖模型

The second architecture is similar to CBOW, but instead of predicting the current word based on the context, it tries to maximize classification of a word based on another word in the same sentence. More precisely, we use each current word as an input to a log-linear classifier with continuous projection layer, and predict words within a certain range before and after the current word. We found that increasing the range improves quality of the resulting word vectors, but it also increases the computational complexity. Since the more distant words are usually less related to the current word than those close to it, we give less weight to the distant words by sampling less from those words in our training examples.

第二種架構類似於CBOW，但是它不是根據上下文來預測當前單詞，而是嘗試根據同一句子中的另一個單詞來最大化單詞的分類。 更準確地說，我們將每個當前單詞用作具有連續投影層的對數線性分類器的輸入，並預測當前單詞前後的特定範圍內的單詞。 我們發現，增加範圍可以提高所得詞向量的質量，但同時也會增加計算複雜度。 由於距離較遠的單詞通常與當前單詞的相關性比與距離最近的單詞的關聯性小，因此我們在訓練示例中通過從這些單詞中抽取較少的採樣來給予距離較遠的單詞較少的權重。