NLP相关阅读

黄军军

Content

1. Word Embeddings: A Survey(2019

2. A Review of Different Word Embeddings for Sentiment Classification using Deep Learning(2018

3. A review of word embedding and document similarity algorithms applied to Academic text(2017

4. WORLD MODELS(2018，Reinforcement learning model)

Word Embeddings: A Survey

Word embedding:词嵌入，就是将文本格式的单词向量化表达，基于某种假设表达成一个具有固定大小的向量。将单词表达为向量可以进行相应地一些运算、距离测度，以及机器学习的算法和策略都可以在上面使用。

自然语言处理的大部分任务几乎都要处理文本或单词，一个好的词嵌入或表达对下游任务的效果至关重要。

2003年Bengio提出了一种神经网络语言模型(NNLM):

在反馈网络其他层之前，首先将原始单词投影到一个所谓的嵌入层上(降维,泛化)

Turian等人（2010）:词嵌入可以在许多NLP任务中用作一个独立的功能

Word Embeddings: A Survey

词嵌入基于的假设：具有相似上下文的单词具有相似的含义，也被称为分布假设(Harris，1954提出)

词嵌入模型
- 基于预测的模型，神经网络+嵌入层，比如word2vec
- 基于计数的模型，统计词共现次数矩阵，比如Glove

文章结构：第2节介绍原始的统计语言建模；
第3节给出词嵌入的综述；
第4节总结，第5节给出一些未来有前景的研究主题

Word Embeddings: A Survey

2.1 The vector space model

当人们试图将分析方法应用于文本数据时，遇到的第一个问题是如何以恰当地方式表示它，并且在相似性等其他运算上满意。早期的方法源于信息检索社区，是Salton 1975年给出的。

他们提出了一种编码程序，集合中的文档由t维向量表示，每个元素代表该文档中包含的不同术语。 这些元素可以是二进制数或实数，可以选择使用加权方案（例如TF-IDF）进行归一化，以解决每个术语提供的信息差异。有了这些向量，就可以计算文档向量之间的相似性(比如使用简单的操作，它们之间的内积)。

Word Embeddings: A Survey

- 2.2 统计语言建模

统计语言模型是在一种语言中，单词分布的概率模型。例如，它们可以用来计算给定单词的下一个单词的可能性(结合上下文)。最早的用途之一是在语音识别领域(Bahl et al.(1983))，以帮助正确识别单词和短语的声音信号(已受到噪音/或错误的通道)。虽然一个完整的概率模型，即包含了所有单词在上下文中出现概率的模型，是不容易的，但是经验上发现，一般小到3个单词的上下文就可以获得一个满意的结果(Goodman(2001)。

一个窗口大小为T的N-gram模型，数学公式如下:

$$P(w_1^T) = \prod_{t=1}^{T} P(w_t | w_1^{t-1}),$$

W_t是第t个单词，W_i^T指w_i到w_T的一个单词序列。该模型的问题是计算的参数空间太大，以及模型无法推广到在训练集中不存在的单词序列。早期有很多改进的工作，然而神经语言模型做的更好。

Word Embeddings: A Survey

- 3. 词嵌入

- 词嵌入是用固定长度的向量表示单词，有很多方法或策略可以做到，本节着重介绍这些方法。
- 词嵌入大体分两类：
- 一类是利用本地的数据，如词的上下文，基于预测的模型，比如之前的神经语言模型。
- 另一类，是利用全部的信息(语料库)，通常是整个存储库范围的统计数据，例如字数和频率，称为基于计数的模型。

Word Embeddings: A Survey

- 3.1基于预测的模型

- 基于预测模型的嵌入发展历史，与神经语言模型(NNLMs)有着深刻联系，前文提到因为它就是将原始单词投影到该网络的第一层，也就是嵌入层。最早的NNLMs是2003年Bengio提出的。
- 尽管早期的结果(通过perplexity得到的评估结果)清楚地表明，神经语言模型确实比之前的基于n-gram的模型更擅长于建模语言，但在阻碍这些模型发展的诸多因素中，经常会提到训练时间过长(有时长达几天或几周)。

Word Embeddings: A Survey

- 3.1基于预测的模型
- Bengio 和Sen`ecal (2003)发现计算代价主要来源于softmax 输出层中的，分配函数和归一化因子。Doucet(2001)等人使用一种重要性采样的概念，绕过了归一化因子的复杂计算,使用辅助分布估计神经网络中的梯度,从词汇表中随机抽样。(重要性采样是蒙特卡洛积分的一种采样策略)。之后没多久，Morin和Bengio(2005)提出了另外一种方法，**层次softmax函数**去降低训练和测试的时间，就是将输出的单词排列成分层的二叉树结构，使用二叉树计算每个单词的概率，即通往该单词节点正确路径的概率。由于二叉树在集合V的高度是|V|/log(|v|)，改进后的计算是指数级的加速（实践中效果不是特别明显）。
- Mnih 和 Hinton (2007)最先提出对数双线性模型（Log-bilinear Model,LBM），他们的工作对后续研究影响很大。Mnih和 Hinton(2008)对之前的模型稍稍做了改进，借鉴了Morin 和Bengio(2005)的想法，对层次softmax轻微修改，提出了所谓的Hierarchical Log-bilinear Model (HLBL)。然而Morin 和Bengio(2005)是从wordnet语料库中预先构建一个单词树，Mnih 和Hinton在2008年特意学习了这种树来解决手边的工作。除了其他较小的优化之外，他们报告了比以前LBL模型(速度快200倍)的巨大改进，并得出结论，使用专门构建的单词树是实现这种结果的关键。

Word Embeddings: A Survey

- 3.1基于预测的模型

- 与之前提到的工作有些相似，Collobert和Weston(2008)从一个稍微不同的角度接近这个问题;他们是最先专门设计模型来学习嵌入。而在先前的工作中，嵌入只是主要工作或者说是语言模型学习的一个有趣的产物。除此之外，他们还引入了两个值得一提的改进:他们使用单词的完整上下文(前后)来预测中心词。最重要的是，他们引入了一个更聪明的方式利用无标签的数据生成一个好的嵌入：他们没有训练语言模型（这不是他们的目标），而是使用错误的或负样本来扩展数据集，然后简单的训练一个模型，该模型可以从负样本中识别出正样本。

Word Embeddings: A Survey

- 3.1基于预测的模型

- 这里，还应该提到米科洛夫的两项具体贡献（2009,2010）。2009年，他提出了一个两步法对NNLM模型自助，从而使用一个单词作为上下文来训练第一个模型。然后，使用第一步结果作为初始的嵌入，在更大的上下文中来训练整个模型。2010年，首次提出了使用递归神经网络来训练语言模型。其论点是RNN将状态保持在隐藏层中，从而有助于该模型记住任意长的上下文，因此无需事先确定要用多少个词作为上下文。

Word Embeddings: A Survey

- 3.1基于预测的模型

- 2012年Mnih 和Teh 给出了一个训练NNLM模型更快的方法，他们是利用的是2010年由Gutmann 和 Hyvarinen提出的一种NCE方法(Noise-contrastive Estimation)，又称为负采样技术。NCE是一种通过对真/假样本的二元决策来估计概率分布的方法。它能够减少NNLMs的训练时间，同时相比于之前的神经语言模型，能获得更好的困惑度值。

Word Embeddings: A Survey

- 3.1基于预测的模型

- 随着2013年Mikolov等人的工作，在NLP社区，词嵌入再次受到关注，并认为是一个值得研究的课题。研究者们分析由递归神经网络模型(2010,Mikolov)训练出的词嵌入，用肉眼去观察在这个编码的向量中可能存在的语法规则。
- 结果发现，他们不仅在数据中发现了语法上的规律，而且还发现了语义上的规律。许多常见的关系，如男女关系、单复数关系等，能够对应于在编码的词向量上进行算术运算(如图1)。

Word Embeddings: A Survey

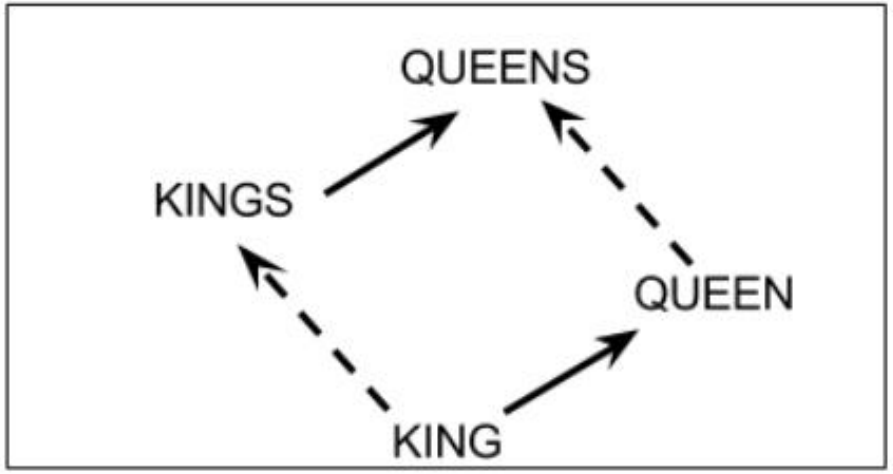


Figure 1: Projection of high dimensional word embeddings (obtained with an RNN language model) in 2D: high-level word embeddings encode multiple relationships between words; here shown: singular-plural (dotted line) and male-female (solid line) relationships. Adapted from Mikolov et al. (2013a).

Word Embeddings: A Survey

- 3.1基于预测的模型

- Mikolov等人(2013年)给出了两格版本，2013b版的CBOW和skip-gram(SG)使用的都是层次softmax(霍尔曼树)，2013c版本则使用的是负采样(用于解决冷僻词的问题)。此外，一些变体提出对频繁单词降采样，以减少由于过于频繁的单词而引起的噪声并加速训练。这些变体在训练时间上表现的更好。

Word Embeddings: A Survey

- 3.1基于预测的模型

- 不久，Mikolov等人(2013年)给出了两个模型用于学习词嵌入，一种叫做continuous bag-of-words(CBOW)，还有一种是 skip-gram (SG) models ，这两种都是对数线性模型(log linear models),而且都使用了两步法(Mikolov,2009)去训练。CBOW和SG模型主要的不同在于更新模型所使用的损失函数。CBOW训练的模型旨在根据上下文预测中心词，而在SG中角色更好相反，中心词被拿来去预测出现在上下文中的每个词。

Word Embeddings: A Survey

- 3.1基于预测的模型

- 在最新的基于预测的词嵌入模型贡献，要提到2016年Bojanowski和Joulin，Fasttext工具包的来源于此，由Facebook提供。他们对SG模型（Mikolov，2013）进行了改进，以此我们学习的不是词嵌入而是n-gram嵌入（它可以组成词）。
- 做出此决定的理由是，事实上在很大程度上依赖于形态和构词构造的语言（例如土耳其语，芬兰语和其他高度虚构的语言）,在单词部分本身中也编码了一些信息，这些信息可用于帮助概括。与SGNS(skip-grap+负采样)相比，他们报告的结果更好（带有负采样的SG变体）（Mikolov et al。（2013c））， 尤其是德语，法语和西班牙语等语言上。下表是基于预测模型策略的综述脉络

Word Embeddings: A Survey

Article	Overview of Strategy	Architecture	Notes
Bengio et al. 2003	Embeddings are derived as a by-product of training a neural network language model.	Neural Net	Commonly referred to as the first neural network language model.
Bengio and Senecal 2003	Makes improvements on the previous paper, by using a Monte Carlo method to estimate gradients, bypassing the calculation of costly partition functions.	Neural Net	Decreased training times by a factor of 19 with respect to Bengio et al. 2003.
Morin and Bengio 2005	Full softmax prediction is replaced by a more efficient binary tree approach, where only binary decisions at each node leading to the target word are needed.	Neural Net, Hierarchical Softmax	Report a speed up with respect to Bengio and Senecal 2003 (over three times as fast during training and 100 times as fast during testing), but at a slightly lower score (perplexity).
Mnih and Hinton 2007	Among other models, the log-bilinear model is introduced here. Log-bilinear models are neural networks with a single, linear, hidden layer (Mnih and Hinton (2008)).	Log-linear Model	First appearance of the log-linear model, which is a simpler model, much faster and slightly outscores the model from Bengio et al. (2003).
Mnih and Hinton 2008	Authors train the log-bilinear model using hierarchical softmax, as suggested in Morin and Bengio (2005), but the word tree is learned rather than obtained from external sources.	Log-linear Model, Hierarchical Softmax	Reports being 200 times as fast as previous log-bilinear models.
Collobert and Weston 2008	A multi-task neural net is trained using not only unsupervised data but also supervised data such as SRL and POS annotations. The model jointly optimizes all of those tasks, but the target was only to learn embeddings.	Deep Neural Net, Negative Sampling	First time a model was built primarily to output just embeddings. Semi-supervised model (language model + NLP tasks).
Mikolov et al. 2013b	Introduces new two models, namely CBOW and SG. Both are log-linear models, using the two-step training procedure. CBOW predicts the target word given a context, SG predicts each context word given a target word.	Log-linear Model, Hierarchical Softmax	Trained on DistBelief, which is the precursor to TensorFlow (Abadi et al. (2015)).
Mikolov et al. 2013c	Improvements to CBOW and SG, including negative sampling instead of hierarchical softmax and subsampling of frequent words.	Log-linear Model, Negative Sampling	SGNS (skip-gram with negative sampling), the best performing variant of Word2Vec, was introduced here.
Bojanowski et al. 2016	Embeddings are trained at the n-gram level, in order to help generalization for unseen data, especially for languages where morphology plays an important role.	Log-linear Model, Hierarchical Softmax	Reports better results than SGNS. Embeddings are also reported to be good for composition (into sentence, document embeddings).

Word Embeddings: A Survey

- 3.2 基于计数的模型
- 前文所述，计数模型它是另外一种词嵌入模型，它不是结合上下文通过算法预测接下来的单词，而是利用语料库中单词上下文的共现次数，通常被表示成单词上下文矩阵(Turney 和 Pantel (2010)。
- 利用单词上下文矩阵来产生单词嵌入的最早的相关例子是Latent Semantic Analysis（潜在语义分析，LSA，Deerwester 1990年），在那里SVD被应用到一个叫做”term-document ”矩阵，它是词矩阵的子集。这种方法最初是为了帮助信息检索而设想的。虽然人们可能对信息检索中的文档向量更感兴趣，但也可以通过这种方式获得单词向量;我们只需要看分解矩阵的行(而不是列)就可以了。

Word Embeddings: A Survey

- 3.2 基于计数的模型

- 不久，Lund和 Burgess 就引入了语言类比超空间(Hyperspace Analogue to Language ,HAL)。
- 他们的策略可以描述如下:对于词汇表中的每个单词，分析它出现的所有文本，计算目标词和每个上下文词之间的共现次数，上下文词到目标词的距离与该计数成反比。在最优的上下文大小为8时，他们获得了比较好的结果。

Word Embeddings: A Survey

- 3.2 基于计数的模型

- 原始的HAL模型未对找到的单词共现计数进行任何归一化处理。因此，非常常见的单词，如定冠词the，对所有与它们同时出现的单词的贡献是不成比例的。Rohde（2006）等人意识到了这个问题，提出了COALS方法，引入了归一化策略来剔除词汇中的这种频率差异。他们认为，与其使用原始计数，不如考虑条件共现，即单词a与单词b共现的可能性比与词汇表中的随机的单词共现可能性大多少。他们用SVD分解共现矩阵相比之前方法获得了更好结果。

Word Embeddings: A Survey

- 3.2 基于计数的模型

- Dhillon（2011）提出了一个稍有不同的方法：Low Rank Multi-View Learning (LR-MVL)。简而言之，它是一个迭代算法，嵌入是通过在给定单词的左右上下文之间利用CCA(Canonical Correlation Analysis，Hotelling 1935)分析来推导的。（典型相关分析）。该模型的一个有趣的特性是，当将嵌入用于下游NLP任务时，它们也会与它们的上下文单词的嵌入连接起来，从而产生更好的结果。相比于矩阵分解法，神经嵌入技术效果上都有所进步。

Word Embeddings: A Survey

- 3.2 基于计数的模型

- Dhillon（2011）提出了一个稍有不同的方法：Low Rank Multi-View Learning (LR-MVL)。简而言之，它是一个迭代算法，嵌入是通过在给定单词的左右上下文之间利用CCA(Canonical Correlation Analysis，Hotelling 1935)分析来推导的。（典型相关分析）。该模型的一个有趣的特性是，当将嵌入用于下游NLP任务时，它们也会与它们的上下文单词的嵌入连接起来，从而产生更好的结果。相比于矩阵分解法，神经嵌入技术效果上都有所进步。

Word Embeddings: A Survey

- 3.2 基于计数的模型

- 基于计数的模型，Lebret和Collobert(2013)，建议将Hellinger pca转换应用于单词上下文矩阵，结果表明比先前的LR-MVL和神经嵌入要好(如Collobert 和 Weston
- 2008，以及2008年Mnih 和 Hinton的HLBL)。

- GloVe （2014，Pennington）：该模型基于以下观点：共现比率对单词对的实际语义信息进行编码(而不是原始计数)，该关系用于对数线性模型导出合适的损失函数，然后对其进行训练以最大化每个单词对的相似度。

Word Embeddings: A Survey

- 3.2 基于计数的模型

- Glove:
- 该相似度通过前面提到的共现比率来衡量每个单词对的相似度。 与其他基于计数的模型以及基于预测的模型（例如SGNS（Mikolov等人，2013c））相比，作者在词类比和NER(named entity recognition)等任务中报告的结果要好。
- **Glove网址链接**
- https://nlp.stanford.edu/projects/glove/

Word Embeddings: A Survey

Article	Overview of Strategy	Notes
Deerwester et al. 1990	LSA is introduced. Singular value decomposition (SVD) is applied on a term-document matrix.	Used mostly for IR, but can be used to build word embeddings.
Lund and Burgess 1996	The HAL method is introduced. Scan the whole corpus one word at a time, with a context window around the word to collect weighted word-word co-occurrence counts, building a word-word co-occurrence matrix.	Reported an optimal context size of 8.
Rohde et al. 2006	Authors introduce the COALS method, which is an improved version of HAL, using normalization procedures to stop very common terms from overly affecting co-occurrence counts.	Optimal variant used SVD factorization. Reports gains over HAL (Lund and Burgess (1996)), LSA (Deerwester et al. (1990)) and other methods.
Dhillon et al. 2011	LR-MVL is introduced. Uses CCA (Canonical Correlation Analysis) between left and right contexts to induce word embeddings.	Reports gains over C&W embeddings (Collobert and Weston (2008)), HLBL (Mnih and Hinton (2008)) and other methods, over many NLP tasks.
Lebret and Collobert 2013	Applied a modified version of Principal Component Analysis (called Hellinger PCA) to the word-context matrix.	Embeddings can be *tuned* before being used in actual NLP tasks. Also reports gains over C&W embeddings, HLBL and other methods, over many NLP tasks.
Pennington et al. 2014	Introduced *GloVe*, a log-linear model trained to encode semantic relationships between words as vector offsets in the learned vector space, using the insight that co-occurrence ratios, rather than raw counts, are the actual conveyors of word meaning.	Reports gains over all previous count-based models and also SGNS (Mikolov et al. (2013c)), in multiple NLP tasks.

Table 2: Overview of strategies for building count-based models for embeddings.

Word Embeddings: A Survey

- 4. 总结

- 在文献中看到的许多建议的进展已经被纳入广泛使用的工具包中，如Word2Vec ，gensim，FastText ，和Glove。
- **gensim网址链接：**
- https://radimrehurek.com/gensim/

Word Embeddings: A Survey

- 5. 未来工作
- 词表征特别是词嵌入研究仍很活跃，这里给出最有前景的几个研究方向：
- （1）为特定于任务的工作调整嵌入
- Maas.(2011), Labutov and Lipson (2013) 及 Lebretand Collobert(2013) 对特定任务嵌入
- （2）基于预测和基于计数的模型之间的联系
- 比如Levy and Goldberg (2014), SGNS模型(Mikolov(2013c))实际上等同于使用一个稍微修改的词上下文矩阵，并使用PMI（pointwise mutual information，逐点互信息）统计进行加权

Word Embeddings: A Survey

- 5. 未来工作
- （3）为更高级的实体组合词嵌入(Composing word embeddings for higher-levelentities)
- 比如给更高级的实体如句子或文档进行嵌入，相关工作有Paragraph2Vec (Leand Mikolov(2014))，Skip-Thought Vectors by Kirosetal.(2015), FastText(Joulin(2016), Bojanowskietal.(2016))

A Review of Different Word Embeddings for Sentiment Classification using Deep Learning

- 本文使用不同的词嵌入策略对亚马逊评论数据集进行情感分类，这些评论数据包含了两种情绪：高兴的和不高兴的。
- 最流行的几种嵌入策略有**skip-gram** method, the **CBOW** model under the word2vec, the **GloVe embedding** method。本文在亚马逊数据集上使用这几种嵌入法作了比较。
- （文章只有3页，介绍部分大同小异，这里只展示比较结果）

A Review of Different Word Embeddings for Sentiment Classification using Deep Learning

- 亚马逊消费者评论数据集有100万个单词，72万个句子。总的数据集被分成70%用于训练，30%用于测试。训练在CPU上，两个epochs完成，每种方法在每个epoch需要花3到4个小时左右。

- 三种方法预测结果如下：结果表明不使用预先训练的权重的嵌入，效果最好。（结果见下表）
- 对消费者的情绪分类，预先训练的权重不能满足要求。另外，这里的结果：**基于词向量的CBOW+负采样法**（基于预测的模型）比Glove（基于计数的模型）预测精度略高。基于词向量的CBOW+负采样>Glove

A Review of Different Word Embeddings for Sentiment Classification using Deep Learning

Embedding w/o pre-trained weights		
Epoch No.	Accuracy(%)	Validation Accuracy(%)
1	94.33	94.64
2	97.60	95.40

GloVe Embedding		
Epoch No.	Accuracy(%)	Validation Accuracy(%)
1	82.07	79.91
2	85.20	83.32

Embedding with Word2Vec CBOW & Negative Sampling		
Epoch No.	Accuracy(%)	Validation Accuracy(%)
1	80.33	82.98
2	85.88	86.53

A review of word embedding and document similarity algorithms applied to Academic text

- 摘要：
- 由于学术文献的数字化和科学基金的增加，学术出版物的速度在过去十年中一直在快速增长。全球阅读模式调查表明，即便是专业的科学家也很难更上步伐。我们提出的长期目标是**创建一个自动工具箱，用于索引、理解和解释所有科学文献**。本文回顾一些词嵌入模型，并介绍针对科学文献的文本相似算法。

A review of word embedding and document similarity algorithms applied to Academic text

- 1. 介绍

- 获取文章现在很便利，一些搜索引擎谷歌学术，web of science...。但是，全球的阅读模式表明，在材料工程或机器学习领域，每天都会遇到几百种新的出版物。正因为如此，每篇文章平均只被阅读5次。

- 2. 基础

A review of word embedding and document similarity algorithms applied to Academic text

- 2.1. 预处理
- 现实世界的文本处理起来很麻烦。语言是高度不一致的，即使在最正式的场合。在本节中，我们将介绍标准的NLP工具集，该工具集用于将文本从原始字符序列转换为更清晰、更易于计算处理的形式。
- 2.1.1 Tokenization
- tokenizationis指将文本拆分为有意义的字符序列组，通常是单词或句子。它几乎总是任何NLP管道的第一步。字符本身很难被计算机解释，但单词大多是独立的语义单位。这是一个很好的抽象层次，很多NLP操作直接作用于单词。
- 简单地按空格分隔并删除任何标点符号(aren' t,co-occurrence,in vitro)、
- NLP库提供了各种特定于语言的标记器，如NLTK

A review of word embedding and document similarity algorithms applied to Academic text

- 2.1.2 Normalization
- 格式和词形变化可能会妨碍匹配、索引或相似性任务。例如，一个语言模型应该能够识别“fox”和“foxes”是相同的名词，或者“have”和“had”是相同的动词。
- 一般来说，规范化是从文本中去除不希望出现的变化，这样轻微的语言差异就不会妨碍对完全相同的概念进行匹配。这通常是通过删除或转换单词的某些部分来实现的，以便只保留一个公共根。然而，规范化会导致信息丢失，这并不总是可取的。
- 第一步显然是小写或大写所有字符，这样首字母大写也能被识别了。然而，去除形态变化是比较棘手：Stemming(词干提取)，Lemmatization(词形还原)

A review of word embedding and document similarity algorithms applied to Academic text

- Stemming （词干提取）
- 用算法方法去除形态变化。大多数语言都有基于词态法转换单词的模式。以复数为例，似乎比较容易规范化，只要去掉后面的“s”或“es”就可以了。确实有很多模棱两可的地方。对于“狐狸”，“es”应该被重新移动，而对于“chocolates”，“s”应该被移除。也有像“语料库”这样的拉丁词有不同的复数“语料库”。这些极端情况并不容易处理，但是可以使用基于规则的方法来实现相当好的基线。当然，它们会导致严重的信息丢失，并且根源可能并不总是正确的。然而，这种简单的实现非常有效，可以很好地扩展到大多数领域。这通常是大规模索引和匹配应用程序的首选标准化方法。

A review of word embedding and document similarity algorithms applied to Academic text

- Lemmatization(词形还原)
- 使用人类编撰的词典从已知单词中提取正确的词元或词根。常用的词汇比如WordNet。这种方法显然比词干提取更准确，但是这种字典的作用是有限的，特别是在特定的文本中。为了最大限度的准确性，**两种方法同时使用**的情况并不少见。

- 2.2 Parsing
- 在隔开和清理单词之后，通常会添加另一个处理层，以便能够在更高的抽象级别上工作。语法解析包括一组注释任务，以便能识别出每个单词相较于文本或上下文中的角色（通常是一个句子）。
- 语法解析通常是需要深入理解的NLP任务的核心。概率语言解析器是20世纪90年代NLP领域的重大突破之一。这样的解析器是第一次启用高质量机器翻译的主要组件之一。

A review of word embedding and document similarity algorithms applied to Academic text

- POS tagging （词性标准）
- 它用一个形态类(如实体类、形容词类或动词类)注释每个标记。一些标记符也更深入，指出了主语的属性，如复数，或动词时态。
- 词性标注通常使用一个参考词典、几个词法启发和一个消歧组件来实现。即使使用字典，单词也可能有不明确的形态类型。例如:“she sat on the back seat(她坐在后座上)”和“he was hit on the back(他被人打了后背)”。消歧是最具挑战性的部分。
- 从历史上看，通过估计同时发生的概率，消除了歧义。最近，最先进的消歧技术已经被使用word嵌入或端到端深度模型的模型所击败。

- Syntactical analysis （语法分析）
- 它是Parsing的第二步。它的任务是确定句子中每个单词的角色，并捕获每个组件之间的依赖关系。

A review of word embedding and document similarity algorithms applied to Academic text

- Syntactical analysis （语法分析）

- 语法分析是一个比词性标注更难的挑战。概率方法，是一个良好的基线，但高质量的结果不能实现，直到最近开始使用深度监督模型。然而，在许多用例中并不需要完整的语法分析。例如，抽取主语-动词-宾语三元组的概率模型也很有效。这样的三元组对于大多数信息提取任务已经非常有用。

- Abstract meaning representation （抽象语义表示）
- 通过命名实体识别和词义消歧将词性标注和句法分析与现实世界信息统一起来，下文将进一步解释。它是我们所拥有的最接近于真正理解文本结构的东西。尽管如此，抽象意义表征仍处于早期阶段。

A review of word embedding and document similarity algorithms applied to Academic text

- 2.3 Word senses
- 意义或概念是语义单位最纯粹的形式。词语和意义是高度相关的，因为在大多数情况下两者之间存在一对一的映射。然而，情况并非总是如此。有些单词有多个义(一词多义)，多个单词可以表示同一个义(同义词)。语言具有高度的模糊性，能够区分和分离感官是达到完全理解的关键。

- Knowledge graphs （知识图谱）
- 知识图是一种很好的语感参考资源。知识图中的每个节点代表一种意义，边定义这些概念之间的关系。知识图，也称为知识库，是一种用结构化的机器可理解的格式编码真实世界信息的方法。
- 有些图表侧重于收集关于世界的公共知识，如Freebase或DBpedia。其他图，如wordnet3，更侧重于区分词的意义和定义概念层次。

A review of word embedding and document similarity algorithms applied to Academic text

- Word sense disambiguation （词义消歧）
- 它的任务是在参考词汇表或知识图中对每个单词进行对应意义注释。词义消歧和词性标注一样，也是一项消歧任务。经典的概率方法依赖于收集语言统计数据，比如共现概率。然后利用这些先验概率将单词与上下文进行比较，并预测最可能的意义。同样的想法也可以用word嵌入来实现。

- Named Entity Recognition （命名实体识别）
- NER是词义消歧的一种应用。NER的想法是对文本进行注释，将单词或单词序列与现实世界的概念联系起来。这对于检索和信息提取非常有趣。有了NER，搜索查询可以使用真实世界的信息，而不是建立在单词索引的基础上。NER有两个部分。首先是识别，有效地匹配词汇和文本，以一种稳健的方式，而不是太敏感的噪音。只有当识别完成后，才会应用消歧。

A review of word embedding and document similarity algorithms applied to Academic text

- Embeddings
- 语义单位的数字表示。最常见的嵌入是word嵌入，词嵌入假设没有歧义，所以它们最终在同一个向量表示中捕获多个意义，这不是理想的。当然，词嵌入可以有效地学习，只要通过扫描大的语料库。有一些感官嵌入算法利用某种形式的聚类来区分感官，无论是在训练期间还是训练后。在某种程度上，word嵌入隐含地将相同的信息编码为知识图。事实上，有一个词和意义嵌入算法的健康生态系统，它使用知识库作为输入数据。
- 2.4 Vector Space Model
- 向量空间模型(VSM)是一种以代数形式表示文档的模型。VSM的基本思想是将文本表示为一个单词的包裹(Bow)。为了有一个紧凑表示，文档中单词的顺序被忽略，文档由单词频率的向量表示。更正式地说，是建立个词汇表。其中对每个单词i，都有一个唯一的整数索引。一个文本d_j被表示成一个列向量v_j，其中v_{ij}储存的是单词W_i在文本d_j中出现的频率。对于标准的VSM模型，Bow被推广成v_{ij}不必是单词出现的频率，而是储存该单词在文本中关联的一个权重。一些比较受欢迎的加权方案有**Tf-idf(逆文档频率)和 BM25**。

A review of word embedding and document similarity algorithms applied to Academic text

- Tf-idf提出的动机是：BoW加权天然地给予出现更频繁的单词赋以更高的权重，但是这些单词不一定更相关，例如stop-words（停用词是指在信息检索中，为节省存储空间和提高搜索效率，在处理自然语言数据（或文本）之前或之后会自动过滤掉某些字或词，这些字或词即被称为Stop Words（停用词）。这些停用词都是人工输入、非自动化生成的，生成后的停用词会形成一个停用词表，比如定冠词a,the等）
- 这些频繁出现的词汇反而掩盖了真正相关的差异。因此，Tf-idf引入了文档频率的概念，它的定义如下。Tf(i,j)表示单词i在文档j中出现的频率，df(i)表示包含单词i的文档数，N表示总的文档数。逆文档频率idf(i)=$\log_2$ (N/df(i))。
- Tf-idf(i,j)=Tf(i,j) *idf(i),这样的定义对于频繁出现的定冠词能赋以一个很低的权。
- tf -idf有一些进一步修改，比如tf和idf被规范化以减少文档大小的影响等。

A review of word embedding and document similarity algorithms applied to Academic text

- BM25 ： 也是一种对Bow加权方案的修改。它保留了tf-idf中的idf,将tf
- 修改成了tf*，tf*的定义为：（依赖于两个超参，默认k=1.75,b=0.75）。

$$tf^* = \frac{tf(k+1)}{k(1-b+\frac{b \cdot DL}{AVDL})+tf}$$

这里DL是文档的长度，AVDL是平均的文档长度。这个公式背后的直觉是，k确保tf*是单调增长且有界的，而b参数化了文档长度被标准化的程度。最终，BM25=tf*.idf

A review of word embedding and document similarity algorithms applied to Academic text

- Document comparison
- VSM模型重要的一个方面。标准的比较度量是余弦相似度，如果向量被归一化，它相当于点积。也可以使用欧几里得距离。还有不太常见的Tanimoto相似度度量，类似于余弦相似度，但有不同的非标准化因子。

- 2.5 Deep Learning (略)
- 2.5.1 Neural networks
- 2.5.2 神经网络优化
- 2.5.3深度学习模式

- 2.6 词嵌入(略)

A review of word embedding and document similarity algorithms applied to Academic text

- 3. 评估框架
- 本章讨论了构建一个框架来回顾语义文档相似性度量的最新进展的过程。这项工作的重点是审查文件相似的媒体科学文本，或更具体地说，同行重新审视学术文献。我们还将主要关注作为相似性度量目标的抽象。3.1训练语料库

- 常用的算法是通过扫描大量的免费文本来学习单词的含义。大多数算法的关键假设是，出现在相似文本中的词具有相似的含义。尽管这个想法从50年代就存在了，Word2Vec (Mikolov, K. Chen, et al., 2013)是第一个有效地使用它来训练单词嵌入的。虽然简单，但这个想法却出奇地有效，正是这一发现开启了“嵌入领域”这个词的诞生。这种简单的共现技术不仅产生了编码相似性或相关性的向量，而且还出人意料地编码了更复杂的语义关系。

A review of word embedding and document similarity algorithms applied to Academic text

- 如上所述，由于缺乏训练数据集，文档相似度模型通常依赖于嵌入所具有的知识挖掘质量。词嵌入已经成为现代自然语言处理的基础之一。
- 大多数学术工作集中在通用词嵌入式和资源，最常见的免费文本来源是**谷歌新闻语料库**和Wikipedia dump。这些来源是如此的标准，有高质量的预先训练的词嵌入可用，而且值得信赖。

- 3.1.1来源和理论基础
- ArXiv ：截至目前约收录有100万条记录，其中80万条是物理文章，约12万条是计算机科学，其余则是数学、化学、生物和经济学等。ArXiv是最大的物理文章数据库，也是仅次于DBLP的第二大计算机科学数据库。它是可用的最佳科学文本来源之一，对摘要的训练产生了相当高质量的嵌入。然而，使用整个数据库引入了太多的噪音，因为在主题上的高差异，即使是80万的物理摘要也无法产生理想的语料库。

A review of word embedding and document similarity algorithms applied to Academic text

- 如上所述，由于缺乏训练数据集，文档相似度模型通常依赖于嵌入所具有的知识挖掘质量。词嵌入已经成为现代自然语言处理的基础之一。
- 大多数学术工作集中在通用词嵌入式和资源，最常见的免费文本来源是**谷歌新闻语料库**和Wikipedia dump。这些来源是如此的标准，有高质量的预先训练的词嵌入可用，而且值得信赖。
- 3.1.1来源和理论基础
- ArXiv ：截至目前约收录有100万条记录，其中80万条是物理文章，约12万条是计算机科学，其余则是数学、化学、生物和经济学等。ArXiv是最大的物理文章数据库，也是仅次于DBLP的第二大计算机科学数据库。它是可用的最佳科学文本来源之一，对摘要的训练产生了相当高质量的嵌入。然而，使用整个数据库引入了太多的噪音，因为在主题上的高差异，即使是80万的物理摘要也无法产生理想的语料库。
- ArXiv OAI-PMH: https://arxiv.org/help/oa/index

A review of word embedding and document similarity algorithms applied to Academic text

- 3.1.1来源和理论基础
- CORE:
- 另一个不错的选择是CORE,开放获取数据库CORE链接
- CORE: https://core.ac.uk/

- PubMed
- 可能是最大的科学开放获取文章数据库。主要从事生物医学和医药领域的研究。名为PubMed Central的开放存取子集包含120万条元数据记录，有趣的是，这些记录中绝大多数包含从原始pdf中提取的高质量全文正文。PubMed已经被鉴定为作为我们当前目的的最佳来源。
- PubMed: https://www.ncbi.nlm.nih.gov/pubmed/

A review of word embedding and document similarity algorithms applied to Academic text

- 3.1.2技术方面
- Data acquisition （数据采集）
- 幸运的是，有一个设计非常好的协议专门用于共享学术出版物的元数据。事实上，这也是大型聚合器(如Crossref、PubMed、DBLP或CORE)能够以相对较低的开发成本统一不同数据库的主要原因之一。（该协议称为OAI-PMH6）
- Parsing (解析
- 主要的障碍是处理模式。OAI-PMH通常处理XML记录，这是相当标准的。但是，如果您的主要数据格式是JSON或类似的dictionary-list嵌套结构，那么就会出现问题，因为转换对于复杂的模式非常重要。(解析PubMed数据非常耗时.

A review of word embedding and document similarity algorithms applied to Academic text

- 3.1.2技术方面
- Cleaning
- 通过将所有内容主体转换为纯文本并将它们连接起来创建了语料库之后，还需要做一些进一步的工作来清理和规范文本。清洗功能是根据经验设计的，通过观察大量的样本和添加规则，直到观察到一个合理的清洁水平。
- 经过一些清理，标准的句子和单词标记符被应用，在NLTK中实现。一些测试使用了lemm，但大多数实现使用wordnet11或类似的KBs作为参考数据12，这在科学媒介中不是很有效。在尝试了几种选择之后，进行归一化也在NLTK中实现。
- Pruning(修剪)
  NLTK: http://www.nltk.org/
- 在训练词嵌入时，通常做一些简单的基于频率的词过滤。非常频繁的单词，如stop-words，没有太多的语义意义，最终会给系统增加噪音。非常不恰当的词语也会引起问题，因为没有足够的样本来正确地训练它们。此外，在科学领域，我们最感兴趣的是已建立的技术词汇，因此，一个很少被提及的词在这种背景下不是很相

A review of word embedding and document similarity algorithms applied to Academic text

- 3.1.2技术方面
- 创建一个中心词汇表也很重要，这样训练和测试数据以及模型就可以正确地同步。我们通过使用来自Gensim库的字典实现来解决这两个任务。它负责对词汇表进行索引和删除。
- 语料库的创建分为三个步骤：
- 1.提取文本字段，渲染到全文并清洗（并行）
- 2.按顺序读取语料库以创建字典
- 3.重新创建语料库，删除词典中未包含的所有单词
- 第一和第二可以同时进行。然而，字典y的实现很可能不是fork-safe的，使用多重处理功能可以大大加快初次通过的速度。

A review of word embedding and document similarity algorithms applied to Academic text

- 3.2 Word embedding evaluation
- 似乎还没有一个明确的词嵌入评价标准。
- **类比法：**
- 最初的Word2Vec文章提出了一个基于他们所称的类比测试的评估。他们创建了一个数据集，每条记录由两对单词组成，这样两对单词之间就有了等价的内部关系。利用向量算法对第二对的预测质量进行了检验，并进行了评价。
- 这种方法确保了嵌入编码复杂的语义，而不仅仅是相似。然而，原始数据集相对较小。其他著名的单词嵌入算法如GloVe也使用了这个数据集，但似乎没有人花时间扩展它。对这种复杂语义关系的进一步研究还表明，该算法可能无法学习人类期望的相同直觉关系(Fu et al.， 2014)。这可能会引入更多的噪声，并降低这种评估的有用性。

A review of word embedding and document similarity algorithms applied to Academic text

- 3.2 Word embedding evaluation
- **Model**
- 将word embeddings作为更复杂模型的输入进行评估，然后评估该模型，这也是一种常见的方法。例如，GloVe pro提出了一个名称实体识别任务进行评价，而分类、情绪分析或聚类也是常用的评价方法。在某些情况下，这可能更容易评估，但它为作者提供了更多的选择，从而导致更多的噪音和不一致性。
- **Similarity datasets**
- 有一个词的相似度和关联度的集合是有用的，已被普遍用于词嵌入评价，最完整的一个是WordSim。WordSim或WS353是非常标准的，在大多数评估过程中使用，但是有一组更小的数据集使用不一致，这使得不同的嵌入算法很难比较。
- 例如，斯坦福的GloVe使用WS353、RG、SCWS和RW。Facebook的FastText ，也使用WS和RW表示英语，但也关注其他主要语言:德语的Gur65、Gur350和ZG222，法语的RG65和西班牙语的WS353。

A review of word embedding and document similarity algorithms applied to Academic text

- 3.2 Word embedding evaluation
- **Similarity datasets**
- 不幸的是，这些数据集相当小。WordSim本身只有353对单词。这些数据集也非常普遍，不包括专门领域的词汇。这也存在了一个相似性和相关性的问题。词嵌入不区分两者和这些数据集。例如，反义词被认为是高度相关但又完全不同的。
- 标准化工作正在进行中。SemEval workshop是统一语义分析领域的一项主要工作，它具有单语和跨语言的相似性任务。然而，在2017年的最后一次迭代中提供的数据集的规模与上面提到的相似。最重要的是，它们处理非常一般的词汇，这项工作的重点是评估科学领域中所有可用的技术。
- **A custom dataset**
- 由于缺乏合适的领域专用词嵌入评价标准，因此决定创建一个自定义的词嵌入评价标准。当涉及到对语义进行编码时，手工构建的知识库与单词嵌入是等效的。

A review of word embedding and document similarity algorithms applied to Academic text

- 3.3 Documentsimilarity evaluation
- 文档相似度的评估比word嵌入的评估更加不一致。这在一定程度上是因为文档相似性属于文档嵌入的范畴，而文档嵌入并不专门用于相似性。相似性度量和文档嵌入是文本分类或情感分析等常见NLP任务的基础。这就是为什么相似模型经常通过把它们输入到更复杂的学习任务中来评估的原因。

- 对分类或情绪的评估要建立得多，而原始相似数据集是稀疏的，因此这使得评估更容易。但是，与word embeddings非常类似，这增加了作者用于评估的选项，从而导致进一步的不一致性。Doc2Vec案例是最著名的文档相似度算法之一，展示了这一问题。

A review of word embedding and document similarity algorithms applied to Academic text

- 3.3 Documentsimilarity evaluation
- Doc2Vec
- 原始的Doc2Vec文章(Quoc V. Le and Mikolov, 2014)提出了三个独立的评价任务:句子情感分析(斯坦福SentimentTreebank Dataset),文档情感分析（IMDB数据集），
- 文档相似性(基于未命名搜索引擎的查询).也有一些第三方对Doc2Vec的评价:(Lau和Baldwin, 2016)和(Dai, Olah, and Quoc V Le, 2015)
- 第一个评价, (Dai, Olah, and Quoc V Le, 2015), 是做定性和定量分析。
- 他们使用t-SNE绘制嵌入图来查看模式，并用两个短语做一些简单的最近邻和类比练习:机器学习和Lady Gaga。定量分析比较全面。他们根据作者的知识创造了172个三元组。他们还创造了至少一个共同的ArXiv类别的20K三元组。最后，又有19,876个三元组从维基百科中提取出来，同样是基于分类。然后将一个简单的预测

A review of word embedding and document similarity algorithms applied to Academic text

- 3.3 Documentsimilarity evaluation
- Doc2Vec
- 第二篇评论文章（Lau和Baldwin，2016年）以“论坛问题重复检测”任务进行评估。他们通过相似性得分对对进行排序，并使用AUC和ROC作为度量。 他们还使用人工注释文档相似性的SemEval数据集。
- SemEva
- 与词嵌入一样，SemEval是标准化语义分析工作的主要驱动力之一。 准确地说，SemEval 2017中的第一个任务是文档相似性任务。 它包含一个称为STS的高质量数据集，其中包含从新闻，媒体字幕和论坛中提取的8,628个句子对。
- 这是一项非凡的工作，也是这一领域中的一种。 但是，这再次不适用于我们的用例。 它着重于句子，而我们试图建立摘要或全文之间的相似性。 而且，像往常一样，它不是特定领域的，我们需要科学领域专有的东西。

A review of word embedding and document similarity algorithms applied to Academic text

- 3.3.1 Sources and rationale
- 这项工作的重点是评估应用于学术出版物的文献相似性。不幸的是，如上所述，没有一个标准的数据集可以用来评估这样的任务。因此，我们着手创建我们自己的一个。
- Data PubMed(Open Access)
- Similarity link
- 要评估文档的相似性，我们需要确定一个属性，该属性将文章链接到表示某种语义关系的其他文章。
- (Dai, Olah,and Quoc V Le,2015) 这个Doc2Vec第三方评估使用来自ArXiv和维基百科的分类创建了一些数据集(Dann, Hauser, and Hanke, n.d)引用(Carpenter and Narin, 1973)提出了以下关于期刊相似性的假设:相似的期刊会有相似的引用模式，相似的期刊会相互引用。受此启发，他们提出了以下假设来评估文档分类和聚类，来自同一期刊的文章是相关的

A review of word embedding and document similarity algorithms applied to Academic text

- 3.3.2 Author link dataset
- 获取作者信息本身也有挑战。在几乎所有的元数据数据库中都可以找到作者的名字，但是这些名字是给定的。
- 作者通常只写他们名字的首字母，他们在整个职业生涯中写名字的时候并不总是一致的。他们的机构信息有时会被提供，这解决了一些案件。然而，以亚洲名字为例，这仍然是一个严重的问题。
- 作者名消歧仍然是一个悬而未决的问题，已开始受到学术界和工业界的广泛关注。

  这也可以求助于PubMed数据库。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1 词向量
- 4.1.1 历史
- (Bengio etal.,2003) 提出一种神经网络语言模型，可以同时学习单词嵌入和统计语言模型。影响很大，基于此的工作被多次提及，例如用一个RNN代替一个简单的前馈网络。
- 然后，直到2013年词向量才提出，Mikolov，K.Chen,提出了两种非常简单的对数线性模型，它超越了所有以前的复杂架构，最重要的是，大大降低了时间复杂性。自2013年文章发表以来，词嵌入已经成为现代深度自然语言处理的基础。当然，已经有很多关于Word2Vec的改进建议，但是经过4年的实践证明，Word2Vec是一个可靠的基准，并且仍然经常作为默认的嵌入式源使用。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.2 Intuition
- Word2Vec (Mikolov, K. Chen, et al., 2013)所依赖的主要假设是:具有相似上下文的单词具有相似的含义。这不是一个新想法，早在(哈里斯，1954年)就提出了。然而，在这个前提下训练一个模型被证明是非常有效的。
- Word2Vec以一组随机初始化的词向量开始。它按顺序扫描语料库，总是在它所查看的每个单词周围保留一个上下文窗口。此时，BoW和Skip- gram模型之间存在一些差异，但是，从本质上讲，该算法计算目标单词和上下文单词之间的点积，并执行随机梯度下降(SGD)最小化这个度量。每当两个词在相似的语境中出现时，它们之间的联系或空间距离就会被加强。在扫描语料库时发现的两个词相似的证据越多，它们就越接近。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.2 Intuition
- 还有最后一个问题要解决。我们刚刚描述的基本模型是为了使向量更接近提供了一个正的强化。对于一个无限的语料库，最小状态是所有的向量都在同一个位置，这显然不是期望的效果。
- 为了解决这个问题Word2Vec提出了Hierarchical Softmax regulator。最后，他们提出了另一种方法，称为负采样。后一种方法负采样更简单，效果也更好。基本想法是每次与目标向量之间的距离最小化，随机抽取几个词，让它们与目标向量的距离最大化。这样，保证非相似词之间的距离（因为是随机抽取的单词，极可能不相似，自然两者距离大）。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.3 数学原理
- 训练Word2Vec时，是用一个固定大小的窗口扫描语料库。这个窗口有个中心目标词，还有一些邻居的单词，邻居的单词又称之为上下文。
- 通常在目标词的两边有相同数量的上下文词。优化由SGD执行，其中每个样本是一个窗口，在目标向量和上下文向量之间定义一个损失函数。
- 原始的Word2Vec 文章(Mikolov, K. Chen,etal.,2013) 提出了两种损失函数:CBOW和Skin-gram。CBOW是连续词袋（Continuous Bag of Words），学习词向量是它是在给定上下文的情况下，预测目标单词。相反，Skin-gram是给定目标词来预测上下文单词.。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.3 数学原理
- 紧跟着(Mikolov,Sutskever,etal.,2013)给出了一个更正式的描述。假定现在有一个语料库，一串单词$w^1;w^2;:::;w^T$，该窗口由参数c定义，其中取目标单词左右两边的c个单词。CBoW中通过对上下文单词求和来预测目标词。最大化目标函数如下：

$$\frac{1}{T}\sum_{t=1}^{T}\log p(w_t|\sum_{-c\leq j\leq c,j\neq 0}w_{t+j})$$

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.3 数学原理
- 相反，Skip-gram中，给定目标单词，预测每个上下文单词都是独立的。

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

- 概率定义成一个softmax，对单词w,u_w是目标词嵌入向量，v_w是上下文词嵌入向量。u^w嵌入被保存，v_w是一个副产品。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.3 数学原理
- 下面的定义是用于skip-gram，对于CBOW需要交换目标词和上下文向量。

$$p(w_c|w_t) = \frac{\exp(v_{w_c}^T u_{w_t})}{\sum_{w=1}^{W} \exp(v_w^T u_{w_t})}$$

- 但是作为一个损失函数，softmax太昂贵了，因为计算梯度的复杂度与词汇量W成正比。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.3 数学原理
- (Mikolov,Sutskever,etal.,2013)提出了两种解决办法。
- 一种是使用Hierarchical Softmax，估计softmax它的复杂度是log(w)。另外一种也是最流行的一种，就是负采样，更正式的说是Noise Contrastive Estimation。每个窗口的负采样目标函数最大化如下：

$$\log \sigma(v_{w_c}^T u_{w_t}) + \sum_{i=1}^{k} \mathbb{E}_{w_i} \equiv P_n(w)[\log \sigma(-v_{w_i}^T u_{w_t})]$$

- k是一个超参数，它指定随机负样本数，与目标和上下文之间的正向作用相反。
- 负样本来自分布Pn(w)，作者发现，使用变换后的一元gram分布$U(w)^{3/4}/Z$效果最佳。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 最后值得一提的细节是单词子抽样。在实践中，仅仅对语料库进行扫描并不能很好地平衡证据。
- 频繁出现的单词，如stop-words (and, the, in…)有非常普遍的意义，但对内容的贡献不大，它们本质上是噪音。然而，这些词汇也有很大的影响，因为它们出现在更多的上下文中。为了避免这个问题，Word2Vec使用一个基于频率的概率，去删除或忽略一些单词。频繁的单词将更可能被忽略，这样它们不会作为噪音破坏平衡。假设wi在语料库的频率是f(wi)，t是一个超参数，则这个概率P(wi)为。

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.4 计算复杂度
- (Mikolov, K. Chen,etal.,2013)对Word2Vec给了一个很详细的复杂度分析，也包括对之前的工作如神经语言模型NNM。我们定义这些算法复杂度如下:
- O=E*T*Q
- E是训练的epochs数目，也就是语料库被扫描的次数;
- T是语料库的大小; Q由每个模型具体而定，用SGD单词更新的复杂度。
- 对于cBoW: Q = C*D + D*log2 (V) ; Skip-gram:Q = C (D + D*log2 (V))
- C是窗口大小也就是上下文大小，D是嵌入的维数，V就是词汇量大小。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.4 计算复杂度
- 在CBOW中我们首先对所有上下文嵌入求平均值，并计算目标和上下文之间的点积，这就转换为计算C个点积，每个点积都有D个浮点运算。C*D也出现在skip-gram，因为也要求上下文与目标的点积。D*log2（V）表示的是Hierarchical Softmax复杂度。skip-gram执行这个操作C次，因为它在技术上要计算每个窗口的C个独立的损失函数(给定目标中心词，预测c个上下文单词)
- C的选择大小通常在[5,20]之间，语料库的大小通常我们的工作大约有200万个不同的标记，它的计算代价是log2(2M)~21.从这可看出Hierarchical Softmax引入对性能提升的重要性。在大词汇量的情况下，负抽样可以更好地工作。我们将用超参数K替代这里的log2(V),K范围设置一般在[3,15].

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.5 Experimental review
- 本节展示Word2Vec应用到我们数据集情况。所有的测试都是在Ubuntu下干净隔离的AWS m4.2xlarge (亚马逊云服务)。分别测试了CBOW和skip-gram，嵌入向量维度100的情形。我们还使用不同大小的语料库执行基准测试，以便我们可以检查算法如何伸缩，包括所需的计算资源和评估的准确性。authors提供的原始实现用于所有基准测试。

- 4.1.5.1 Computational benchmark
- 图4.1和4.2显示cBoW和skipgram的训练时间和最大内存。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms

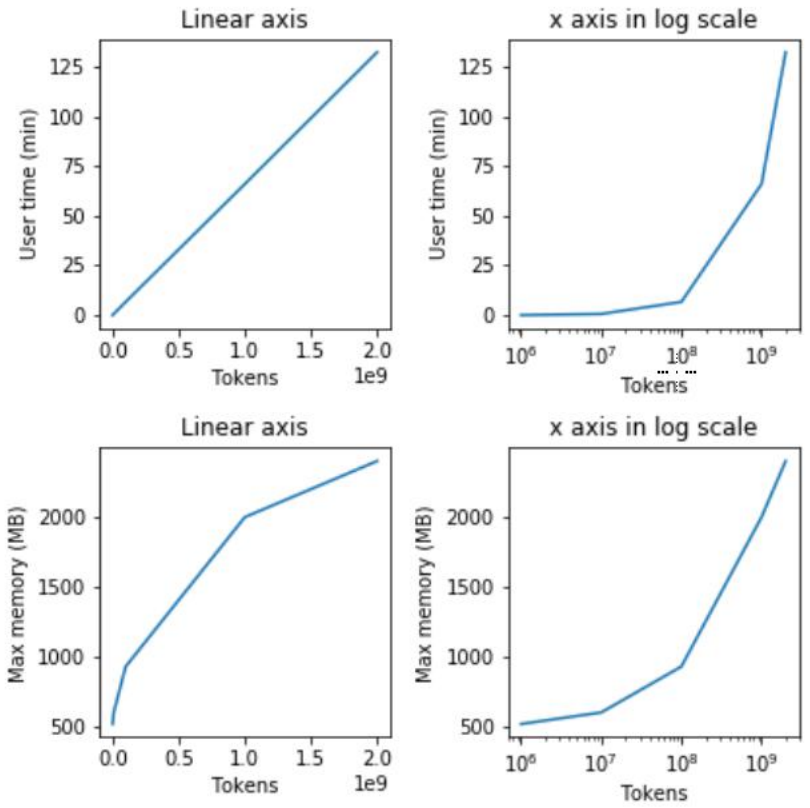


FIGURE 4.1: Computational benchmark of cBoW

执行时间显然是线性伸缩的，而且惊人地清晰。每个实例训练3次，结果一致。然而，这是有道理的。该算法需要一次遍历来构建一个字典，然后根据所选择的epoch的数量进行若干次遍历。对语料库中的每个窗口和单词执行相同的更新操作。Word2Vec也是在干净的c++上实现的，具有最小的依赖性，因此不存在噪声开销和明显的伸缩性也就不足为奇了。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms

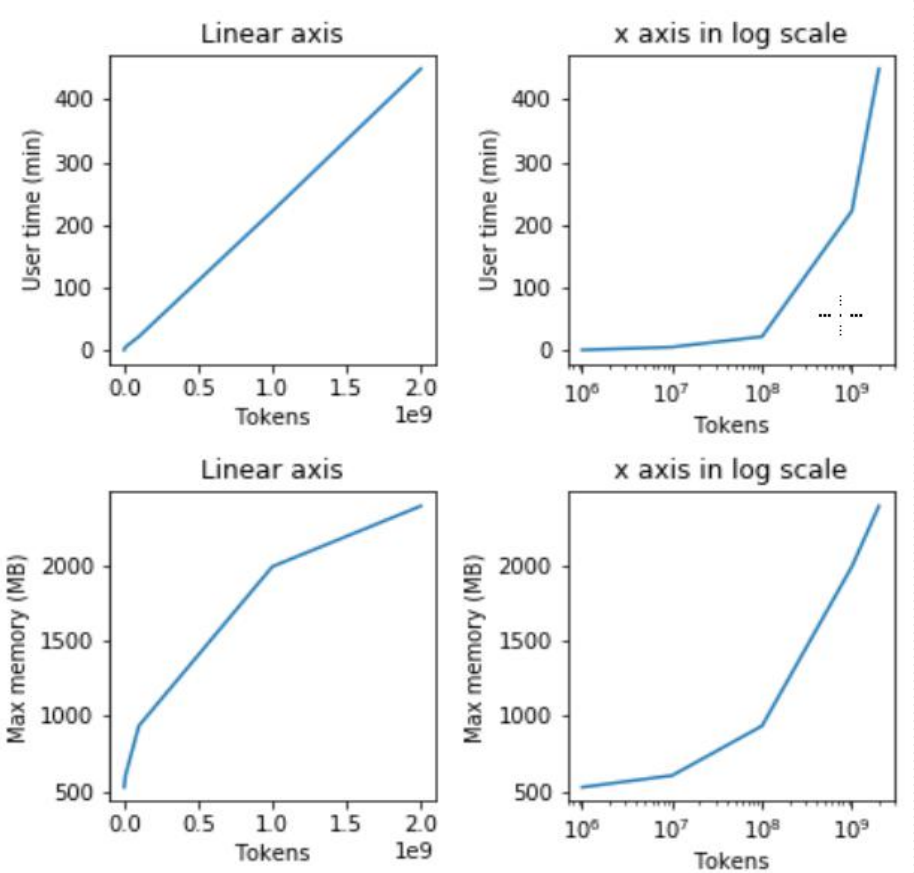


FIGURE 4.2: Computational benchmark of skip-gram

在这两种情况下，内存需求似乎都以O(sqrt(n))为单位进行伸缩。因为大部分内存是由嵌入的词汇表消耗的。当我们探索更大的语料库时，词汇量会增加，但是在更大的范围内，新术语会越来越少。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.5.2 评估
- 从图(4.3和4.4)可以看出，相对于训练语料库中标记的数量，单词嵌入的准确性是如何变化的。通过将单词嵌入与UMLS(医学语言系统)三元组进行比较来进行评估.值得一提的是，对于较小的语料库，测试词中有很大一部分是模型未知的。 通过忽略具有未知单词的任何三元组，我们同时包括了总的准确性和准确性。

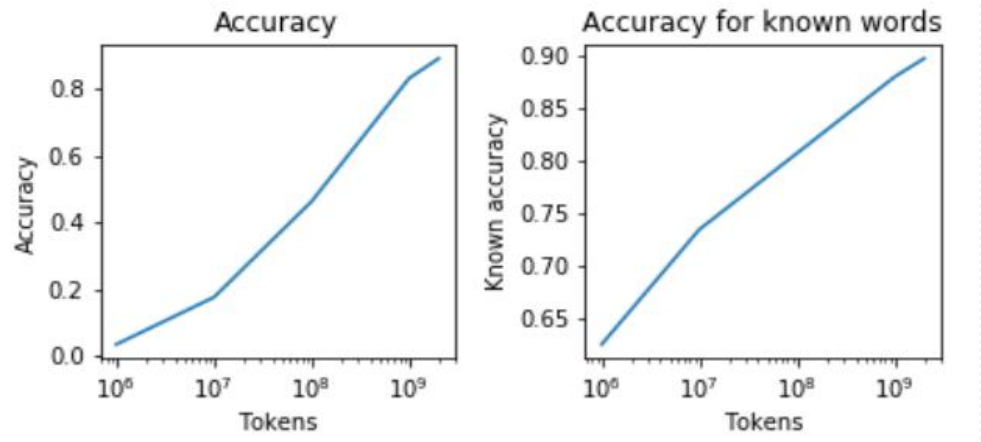


FIGURE 4.3: Evaluation of cBoW

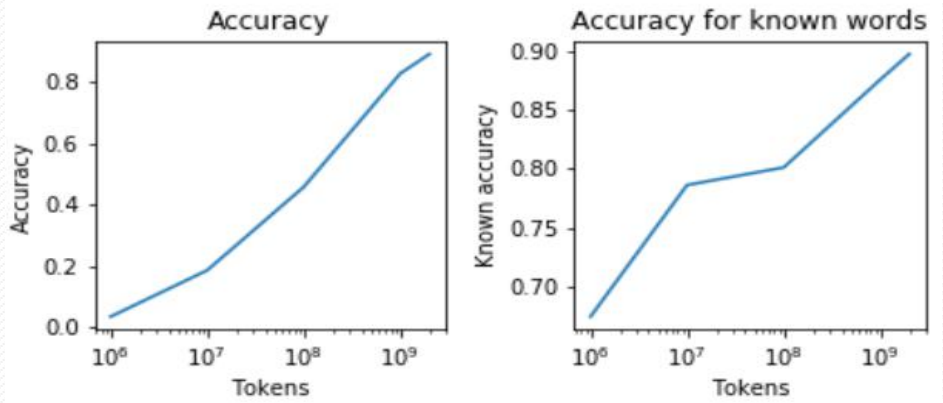


FIGURE 4.4: Evaluation of skip-gram

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.1.5.2 评估
- 这两个模型的结果也非常相似。在这两种情况下，结果似乎都有对数增长。孤立地看，Word2Vec对于科学词汇的性能似乎非常出色，即使没有超参数调优也是如此。
- 考虑到训练集和测试集的抽取是来自不同的数据源，3649个测试三元组，也就是7298个比较。在这种情况下，90%的准确率是值得注意的。更精确的正确率表格如下：Skip-gram表现稍微好些。

Word2Vec evaluation	1M	10M	100M	1B	2B
cBoW - Total	0.03	0.17	0.46	0.83	0.89
Skip-gram - Total	0.04	0.18	0.46	0.83	0.89
cBoW - Known words	0.67	0.73	0.80	0.85	0.90
Skip-gram - Known words	0.67	0.79	0.80	0.88	0.90

TABLE 4.1: The Word2Vec evaluation accuracies by the number of tokens used for training, including total accuracy and known word accuracy.

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2 Glove
- Glove(Pennington, Socher, and C. Manning, 2014)可以说是仅次于Word2Vec的第二大著名的词嵌入算法。它是同一概念的不同理解，Glove和Word2Vec是一枚硬币的两面，虽然它们表面上没什么相似。正因为如此，两者目前被认为是完全等价的。他们在大多数任务中表现相似，当然目前普遍的看法似乎是Glove训练稍微快一些。
- 4.2.1 直觉
- 在原始的文章(Pennington, Socher, and C. Manning, 2014)中，作者提出了两种用于词嵌入训练的模型族：global matrix factorization methods(如LSA)和local context window methods(如Word2Vec)。作者声称两者都有明显的缺陷，比如像Word2Vec利用上下文窗口方法，它没有用到全局的词共现统计的信息。GloVe的主要动机是找到一个中间立场：该算法作用于全局统计信息，但实现的向量空间语义结构与Word2Vec相同。 GloVe还通过明确属性共现概率来阐明创建此类语义矢

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2 Glove
- 它们的主要相似性度量是共现概率。通过一个例子可以更好地理解这一点，原文给出了一个很好的例子。考虑两个相关的单词i=ice和j=stream。我们可以通过观察这些词与一组k个探查词的共现来检查它们之间的关系（P_{ik}/P_{jk}）.
- 由i和j构造一个语义轴，如下：

Probability and Ratio	k = solid	k = gas	k = water	k = fashion
$P(k\|ice)$	$1.9x10^{-4}$	$6.6x10^{-5}$	$3.0x10^{-3}$	$1.7x10^{-5}$
$P(k\|steam)$	$2.2x10^{-5}$	$7.8x10^{-4}$	$2.2x10^{-3}$	$1.8x10^{-5}$
$P(k\|ice)/P(k\|steam)$	8.9	$8.5x10^{-2}$	1.36	0.96

TABLE 4.2: From the original paper (Pennington, Socher, and C. Manning, 2014)

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2 Glove
- k = solid时与目标词语义上很相关，而k = gas则刚好相反相关很小。另外还有一个不相干的词k = fashion二者比值会表现接近1，k = water虽然是与目标词高度相关的，但比值也接近1是因为用它来表达这两个单词的关系是多余的。
- GloVe将上述现象正式化，并对嵌入式进行训练，使其能够模拟这样的结构。我们认为，GloVe和Word2Vec之所以表现得如此相似，是因为它们本质上是在优化相同的目标。它们都是在假定语境相似的词具有相似的意义的前提下产生的。
- GloVe的作者最初声称Word2Vec不会利用全局统计信息，但是顺序扫描语料库确实隐式捕获了这些统计信息，因为更多的相似性证据进一步加深了嵌入之间的距离。GloVe使用显式概率进行处理，而Word2Vec通过实际遇到具有不同频率的文本结构来进行处理。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2 Glove
- 4.2.2 数学原理
- 在上文中构造了两个词i=ice,j=stream和一组词构成的语义轴的关系，这个相关概念也可以用如下的方程表示：

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

- 这个还没定义的F函数方程可做一些简化，用作差和点积替代。

$$F((w_i - w_j)^T \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2 Glove
- 4.2.2 数学原理
- 假设F是同态的，在(R ； +) and （ R ； X ）之间，那么F只能是指数函数

$$F((w_i - w_j)^T \tilde{w}_k) = \frac{F(w_i^T \tilde{w}_k)}{F(w_j^T \tilde{w}_k)}$$

$$F(w_i^T \tilde{w}_k) = P_{ik} = \frac{X_{ik}}{X_i}$$

$$w_i^T \tilde{w}_k = \log P_{ik} = \log X_{ik} - \log X_i$$

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2 Glove
- 4.2.2 数学原理
- Xi是单词i的总的频率，X_{ik}是第i个单词在词k的上下文出现的计数。
- 进一步简化，log(Xi)可看做不依赖与第k词，因此可以用偏置bi替代log(Xi)，为了对称，再加一个b

$$w_i^T \tilde{w}_k + b_i + \tilde{b_k} = \log X_{ik}$$

- $F(W_i^T * Wk)=P_{ik}$暗含了这个点积的优化与这两个词共现的概率相关，这就和word2vec联系起来。这个方程已经很简化了，但是还有需要改进的地方，第一当i和k没有共现时$\log(X_{ik})$会发散。可以将log(x)改为log(1+x)；第二，非常频繁的词共现也是不太相关的(stop words)，主要导致噪音增加，我们可以用一个权函数$f(X_{ij})$对这种现象做一个补偿。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2 Glove
- 4.2.2 数学原理
- F函数的选择有很多，但是作者都使用如下这样一个函数：

$$f(x) = \begin{cases} (x/x_{max})^{\alpha} & \text{if } x < x_{max} \\ 1 & \text{otherwise} \end{cases}$$

- 选择X_{max} = 100，经验上a=3/4是比较适当的，巧合的是这种尺度对于word2vec也是最佳的。最后把这个转为最小二乘优化问题，就是最终的Glove模型：

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_k + b_i + \tilde{b_k} - \log(1 + X_{ik}))^2$$

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2.3 计算的复杂性
- 经典的基于窗口的方法如word2vec，取决于语料库的大小；而Glvoe则是取决于词汇量大小。这是因为Glove要保存一个共现矩阵，这个共现矩阵是所有的词对出现的次数，然后用这个矩阵训练。因此，Glvoe计算复杂度上界就是O(V2)，V是词汇量大小。有一个词嵌入算法的计算规模与词汇量相关是非常方便的，
- 因为，词汇量在更大的语料库中是停止增长的。由于这一点，我们可以使用任意大的语料库来进行任意精确的共现统计。然而，真实世界的词汇表通常有数十万个单词，这将增加数千亿的复杂性，这比几乎所有当前的语料库都要大。
- 如果，共现的窗口足够小，那么共现矩阵就是稀疏的。因此，复杂度将取决于矩阵的非零元素。

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.2.3 计算的复杂性
- 作者利用以下两个假设来逼近矩阵中非零元素的个数：
- 1.语料库中的单词数量与共现矩阵中所有元素的总和成正比。显而易见，语料库大小应该是总和的两倍。
- 2.两个词的共现可以用该单词对的频率等级的幂律函数来近似。语言统计中的幂律模式是一个观察到的现象，这种建模是NLP中的标准。
- 从这些假设出发，作者得出了一个非零元素模型：
- 得出结果Glove的复杂度是$O(|C|^{0.8})$，其中$|C|$是语料库的大小。
- 这解释了Glove为什么比word2vec训练快些，因为word2vec复杂度约为$O(|C|)$.

A review of word embedding and document similarity algorithms applied to Academic text

- 4.2.4 实验
- 就像Word2Vec一样，我们可以清楚地看到Glove对语料库大小的线性缩放。在时间和内存上，Word2Vec和GloVe的缩放模式几乎是相同的。然而，GloVe需要大约两倍的内存和几乎六倍的处理时间。这与学术界普遍认为GloVe等同于或略快于Word2Vec的说法正好相反。

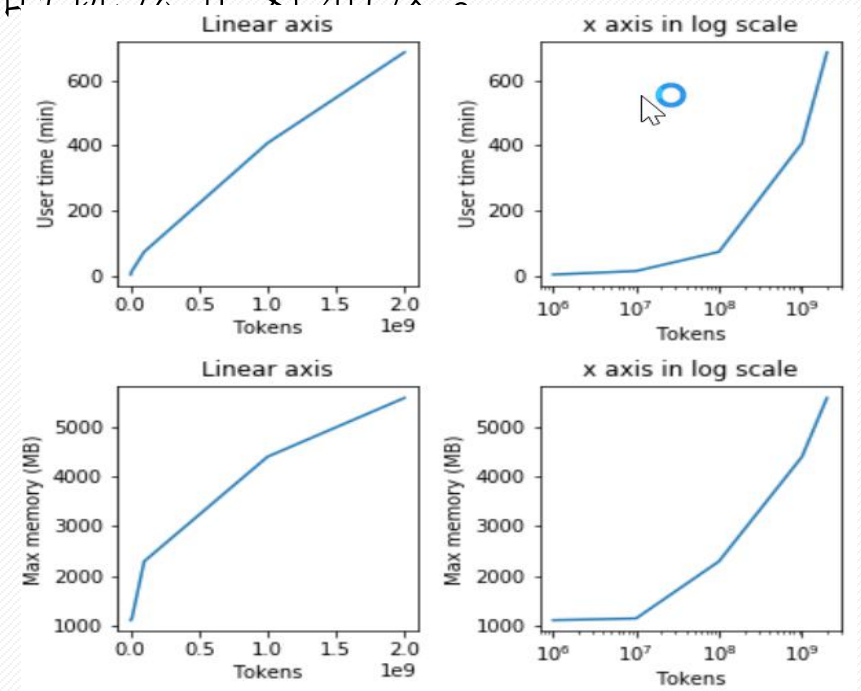


FIGURE 4.5: Computational benchmark of GloVe

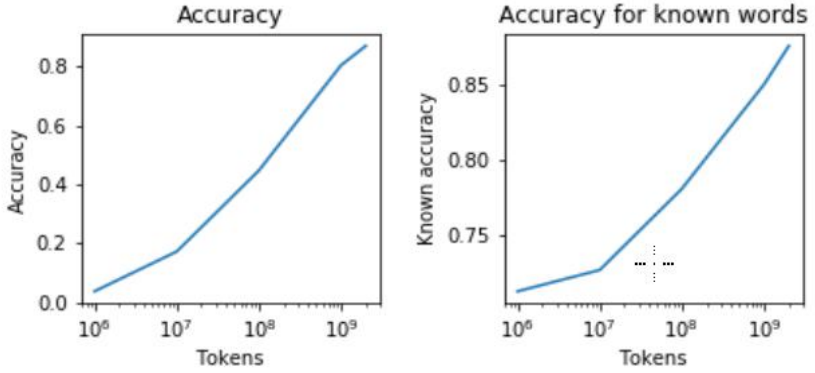


FIGURE 4.6: Evaluation accuracy of GloVe

GloVe evaluation	1M	10M	100M	1B	2B
Total accuracy	0.04	0.17	0.45	0.80	0.87
Known words	0.71	0.73	0.78	0.85	0.88

TABLE 4.3: The GloVe evaluation accuracies by the number of tokens used for training, including total accuracy and known word accuracy.

A review of word embedding and document similarity algorithms applied to Academic text

- 4. Review of word embedding algorithms
- 4.3 Fast Text ...
- 4.4 Word Rank ...

Word embeddings	1M	10M	100M	1B	2B
W2V cBoW - Accuracy	0.03	0.17	0.46	0.83	0.89
W2V cBoW - Known	0.67	0.73	0.80	0.85	**0.90**
W2V Skip-gram - Accuracy	0.04	0.18	0.46	0.83	0.89
W2V Skip-gram - Known	0.67	0.79	0.80	0.88	**0.90**
GloVe - Accuracy	0.04	0.17	0.45	0.80	0.87
Glove - Known	0.71	0.73	0.78	0.85	0.88
FastText - Accuracy	**0.81**	**0.88**	**0.90**	**0.93**	-
WordRank - Accuracy	0.02	0.21	0.45	0.78	0.89
WordRank - Known	0.69	0.75	0.77	0.84	**0.90**

TABLE 4.6: All results from word embedding evaluation unified.

WORLD MODELS

- (David Ha, Jurgen Schmidhuber,2018)
- 摘要:
- 建立了一个生成神经网络模型world models。它可以在无监督的情况下快速训练，针对时间和空间环境学习一个压缩的表示。通过使用world models抽取到的特征，传入到一个代理（agent），我们能训练一个简单紧凑的策略解决问题。
- 这篇文章的交互式版本可以在下面链接获取到:
- https://worldmodels.github.io

- 1.介绍
- 人类根据有限的感知能力来开发世界的心智模型。我们的决策，行为都是基于这种内在的模型。系统动力学之父（Jay Wright Forrester）将这种心智模型描述为:

WORLD MODELS

The image of the world around us, which we carry in our head, is just a model. Nobody in his head imagines all the world, government or country. He has only selected concepts, and relationships between them, and uses those to represent the real system. (Forrester, 1971)

为了处理我们周边大量的信息，我们的大脑基于时间和空间的信息学习到了一个抽象的表达。我们能够观测一个场景并记住这个抽象的描述（(Cheang&Tsao,2017;Quirogaetal.,2005)）。

有证据表明我们在任何瞬间感知到的，都是我们的大脑基于内在模型对未来做出的预测所决定的。(Nortmannetal.,2015;Gerritetal.,2013)。

WORLD MODELS

- (David Ha, Jurgen Schmidhuber,2018)

- 其中一种理解我们大脑内在预测模型认为：它可能不仅仅一般性的预测未来，而是结合当前机动行为，预测未来的感知数据(Kelleretal.,2012;Leinweberetal.,2017) 。

- 基于这个预测模型，比如当我们面临危险的时候，我们能够本能地、快速做出反应，而不需要有意识的计划一个行为(Mobbsetal.,2015)。拿打棒球举例。击球手只有毫秒的时候决定如何挥动球棒，这要比视觉信号到达大脑所需的时间更短。我们之所以能够打出每小时100英里的快球，是因为我们能够凭直觉预测球的去向和位置。对应职业运动员，这些都是潜意识发生的，或者是肌肉反应。

WORLD MODELS

- (David Ha, Jurgen Schmidhuber,2018)
- 1.介绍

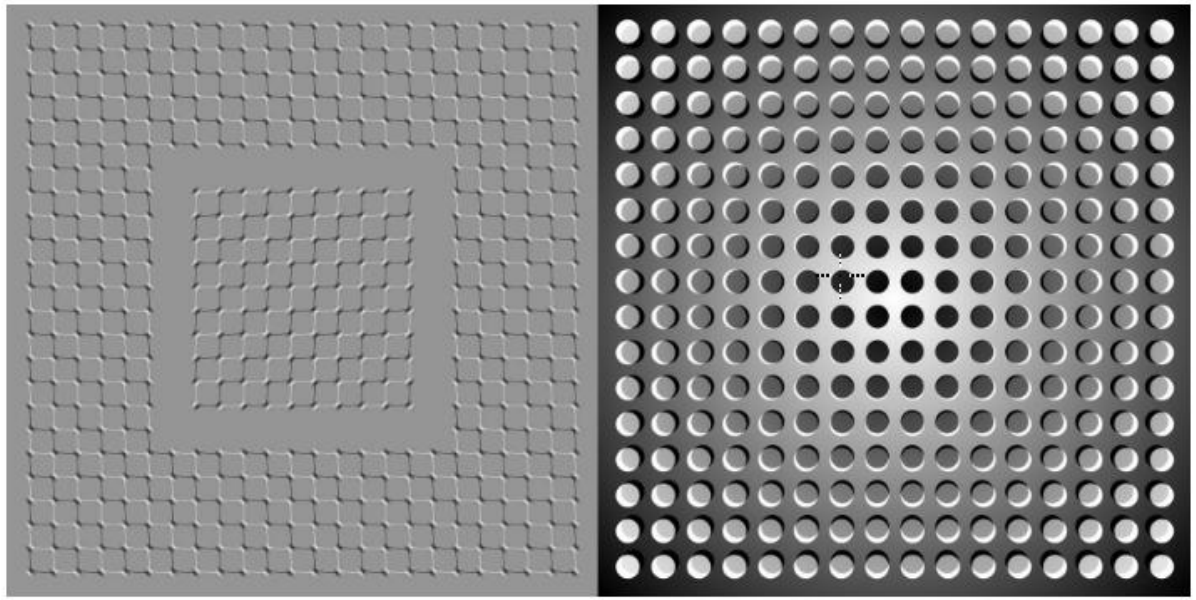


Figure 2. What we see is based on our brain's prediction of the future (Kitaoka, 2002; Watanabe et al., 2018).

WORLD MODELS

- 1. 介绍
- 在许多强化学习问题中，智能代理都得益于对过去和现在状态的一个好的表达，而一个功能强大的预测模型最好是通用的，比如RNN表现就不错。
- 大型的神经网络是高度表达的模型，可以学习数据的丰富的空间和时间表示。然而，文献中的许多RL方法往往只使用参数较少的小规模的神经网络。另外RL算法常常受到信用分配问题的瓶颈，这使得传统的RL算法很难学习一个大模型的数百万个权值，因此在实践中，使用较小的网络，因为它们在训练中迭代速度更快，从而获得一个好的策略。
- 我们希望能够有效地进行大规模训练基于RNN的代理。通过将代理分为一个world模型和一个小的控制模型。我们首先训练一个大型神经网络，再以无监督方式学习world model，然后训练较小的控制器模型，以使用world model执行任务。

WORLD MODELS

- 2. Agent model
- 受我们自己认知系统的启发，我们提出一个简单的模型。在这个模型中，代理有一个视觉感知的组件，它能够将看到的压缩成一个很小的代码表达。另外，基于历史信息，代理有一个记忆组件用于对未来做预测。最后，基于视觉和记忆组件创建的表达，代理还有一个做出决策的组件。

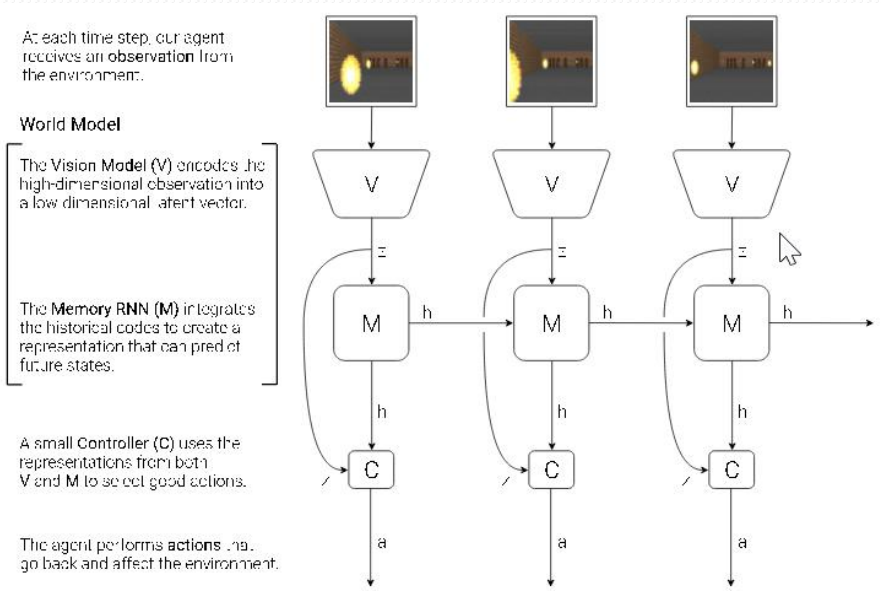


Figure 4. Our agent consists of three components that work closely together: **Vision (V)**, **Memory (M)**, and **Controller (C)**

WORLD MODELS

- 2. Agent model
- 2.1. VAE (V) Model
- 每一步时间，环境都会给代理一个高维的输入，这个输入经常是一个2维图像帧，它是一个视频序列的一部分。V 模型就是学习一个抽象的表示，针对这些图像帧。
- 这里选择VAE(Variational Auto encoder ,(Kingma& Welling, 2013; Rezende et al., 2014))
- 作为V模型，将

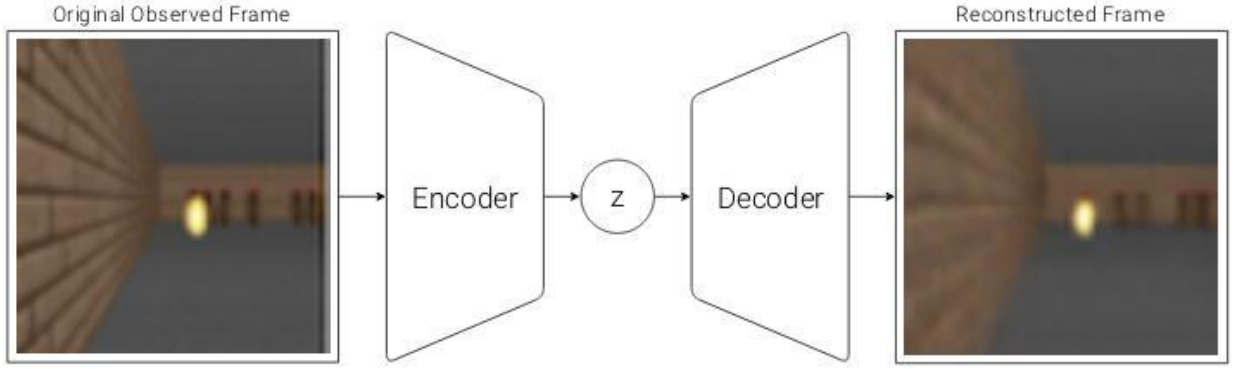


Figure 5. Flow diagram of a Variational Autoencoder (VAE).

WORLD MODELS

- 2. Agent model
- 2.2. MDN-RNN (M) Model
- M模型是预测未来的时间V模型可能产生的Z向量。因为现实环境很复杂，且随机的，所以这里预测未来的Z是一个概率密度函数P(z)而不是确定性的。

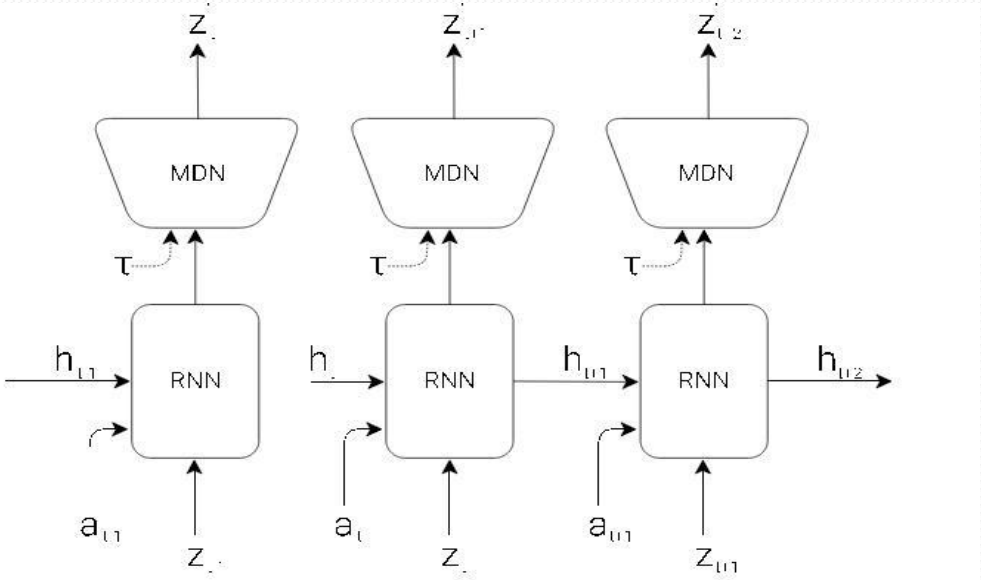


Figure 6. RNN with a Mixture Density Network output layer. The MDN outputs the parameters of a mixture of Gaussian distribution used to sample a prediction of the next latent vector z.

WORLD MODELS

- 2. Agent model
- 2.2. MDN-RNN (M) Model
- M模型是预测未来的时间V模型可能产生的Z向量。因为现实环境很复杂，且随机的，所以这里预测未来的Z是一个概率密度函数P(z)而不是确定性的。

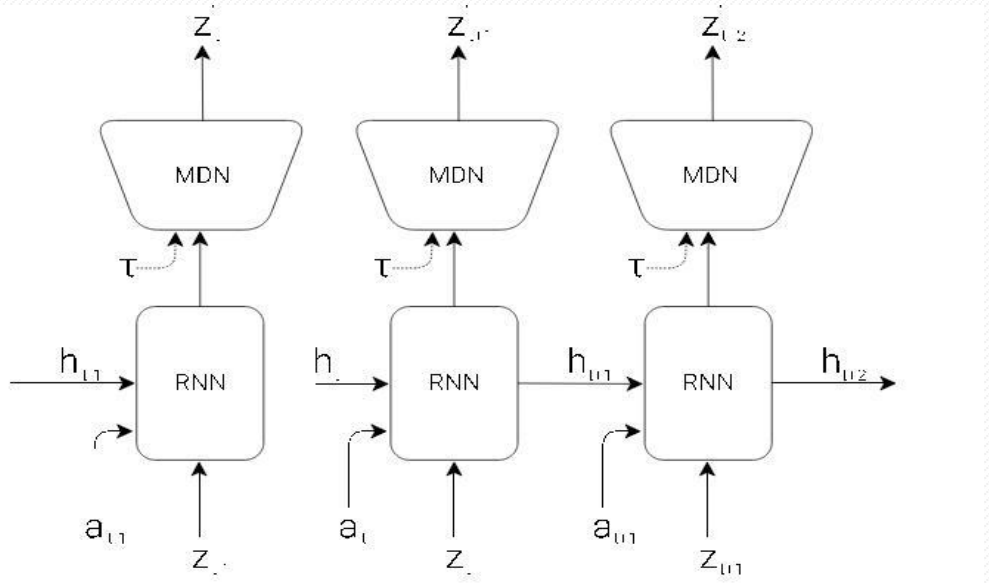


Figure 6. RNN with a Mixture Density Network output layer. The MDN outputs the parameters of a mixture of Gaussian distribution used to sample a prediction of the next latent vector z.

这里用高斯混合分布逼近这个p(z),给定当前和过去信息，训练RNN输出下一时刻Z(t+1)。

RNN模型训练即：$P(z_{t+1} \mid a_t, z_t, h_t)$

a_t是t时刻采取的行为，h_t是RNN在t时刻的隐藏状态，z_t是t时刻潜在向量。

WORLD MODELS

- 2.Agent model
- 2.2. MDN-RNN (M) Model
- 在采样期间，我们设置一个温度参数，用于控制模型的不确定性，类比(Ha & Eck,2017)的做法。稍后我们会发现调整对训练我们的控制器很有用。
- 这种方法就是熟知的结合RNN的Mixture Density Network，MDN-RNN(Graves,2013; Ha,2017a)，
- 该方法已经被用于序列生成问题如生成手写字(Graves,2013），或生成素描(Ha & Eck,2017）。

WORLD MODELS

- 2. Agent model
- 2.2. MDN-RNN (M) Model

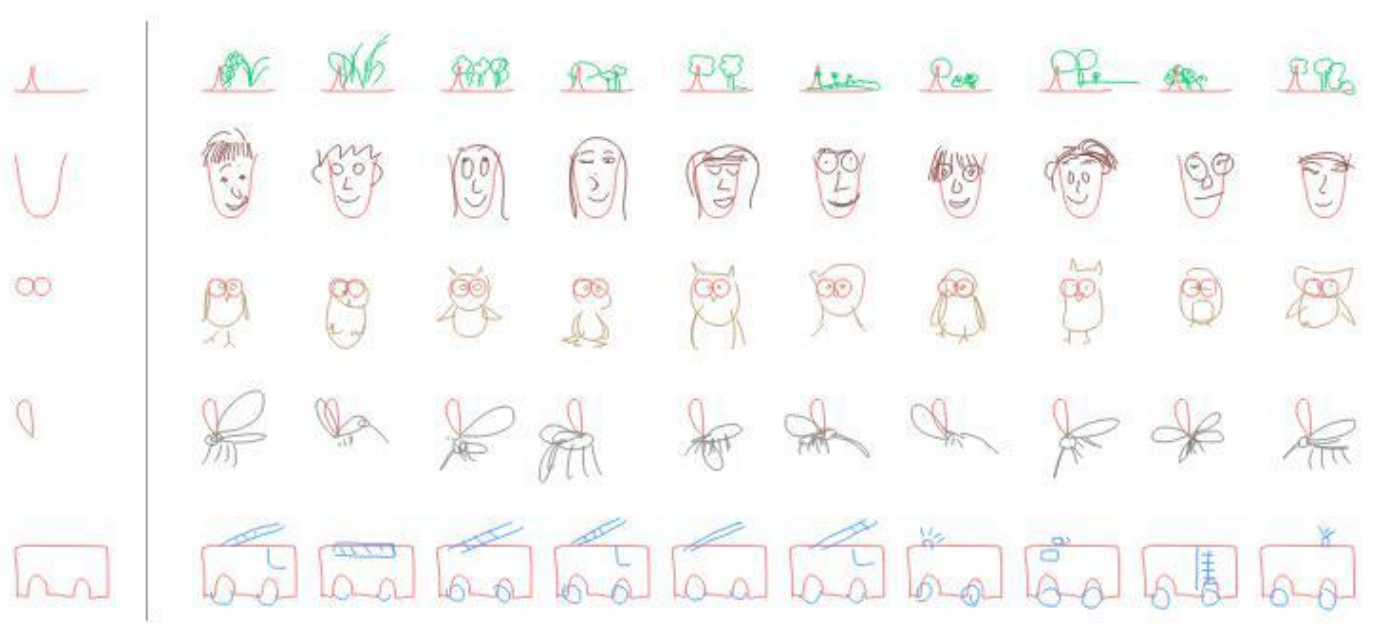


Figure 7. SketchRNN (Ha & Eck, 2017) is an example of a MDN-RNN used to predict the next pen strokes of a sketch drawing. We use a similar model to predict the next latent vector z_t.

WORLD MODELS

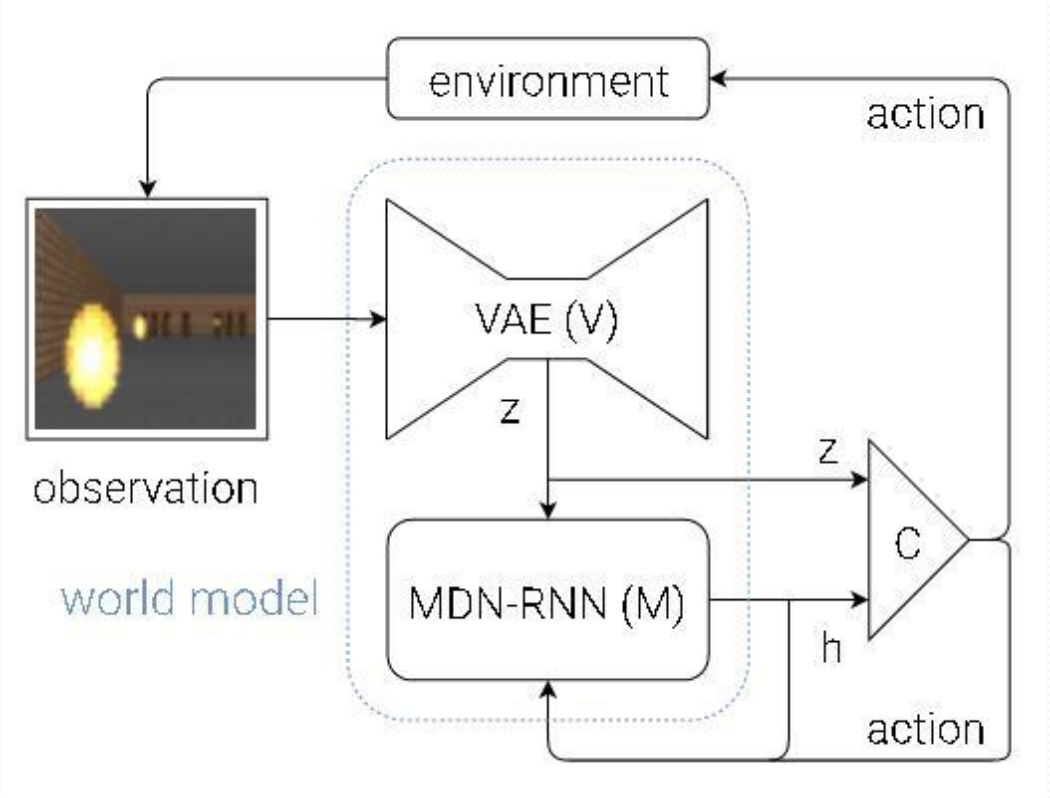
- 2.Agent model
- 2.3 Controller(C) Model
- 控制器模型负责确定要采取的措施，以便在环境部署期间让代理的预期累积回报最大化。
- 在我们的实验中，我们故意让C尽可能的小和简单，并与V和M分开进行训练，以便我们代理的大部分复杂性都位于world模型（V和M）中。
- C选择为一个简单的线性层，每个时间步将Z_t，h_t映射到行为a_t：

$$a_t = W_c \left[z_t \; h_t \right] + b_c$$

- W_c权矩阵，b_c偏置，输出为行为向量a_t。

WORLD MODELS

- 2.Agent model
- 2.4 V+M+C组合
- 下面流程表展示的是V,M，C是怎样与环境交互的。



WORLD MODELS

- 2.Agent model
- 2.4 V+M+C组合
- t时刻输入图片，经过V处理得到一个z_t，将z_t和M中的隐藏状态h_t输入到C中，
- C输出一个行为向量a_t，它将影响环境。然后M把当前的z_t和C产生的a_t做为输入，更新自身的隐藏状态产生h_{t+1}，用于下一时刻t+1迭代。

- 下面是代理模型的伪代码，在OpenAI Gym环境 (Brockman et al., 2016)被使用。

WORLD MODELS

- 2. Agent model
- 2.4 V+M+C组合

```
def rollout(controller):
  ''' env, rnn, vae are '''
  ''' global variables  '''
  obs = env.reset()
  h = rnn.initial_state()
  done = False
  cumulative_reward = 0
  while not done:
    z = vae.encode(obs)
    a = controller.action([z, h])
    obs, reward, done = env.step(a)
    cumulative_reward += reward
    h = rnn.forward([a, z, h])
  return cumulative_reward
```

WORLD MODELS

- 2.Agent model
- 2.4 V+M+C组合
- 选择一个较简单的控制器C（当然损失函数得是可微的），V和M使用gpu加速，反向传播去训练，这样我们的模型复杂性和更多的参数都保存在V和M中，线性模型C相比较而言参数较小。这种选择让我们可以探索更多非传统的方式来训练——
- 例如evolution strategies，(ES,Rechenberg,1973;Schwefel, 1977)去处理更复杂的强化学习任务。为了优化C的参数，我们采用Covariance-Matrix Adaptation Evolution Strategy(CMA-ES)((Hansen, 2016; Hansen & Ostermeier,2001))，这是因为它在最多有几千个参数的解空间中工作得很好。
- 我们在一台计算机，多个cpu并行优化这个控制器C的参数。
- （更多模型信息和训练过程，以及使用环境可参考附录章节）。

WORLD MODELS

- 3.赛车实验
- 在本节中，我们将描述如何训练前面建立的代理模型来解决赛车任务。据我们所知，我们的代理是已知的、第一个达到该任务所需分数的解决方案。
- 3.1. World Modelfor Feature Extraction
- 通过模型提取的抽象特征，输入到控制器C中，接着训练C去做一连串的控制任务。比如在一个自顶向下的赛车环境CarRacing-v0 (Klimov,2016)，学习如何从输入像素到导航驾驶。

- 在这个环境中，赛道是随机生成的，我们的代理会因为在最短的时间内访问了尽可能多的瓷砖而获得奖励。驾驶员控制三种连续动作:向左/右转向、加速和刹车。

WORLD MODELS

- 3. 赛车实验
- 为了训练我们的V模型，我们首先收集1万个随机滑行的环境数据集。
- 首先我们让代理随机行动，对环境进行多次探索，并记录随机的行动a_t和由此产生的环境观察。然后我们使用这个数据集训练V学习每一帧观察的潜在空间Z。训练我们的VAE，将每一帧图片编码成低维潜向量z，方法是最小化给定帧与由z解码器生成的重构图片之间的差异。然后我们使用已经训练的V模型，去预处理每一帧图像，转为z_t，去训练M模型，使用预处理的数据z_t，以及随机记录的行为a_t，然后MDN-RNN就可以训练了。
- 即训练 $P(z_{t+1} \mid a_t, z_t, h_t)$ ，视为一个混合高斯分布。
- 注：原则上，我们可以对两种模型进行端对端训练，虽然我们发现分别训练更实际，也能取得令人满意的效果。在单个GPU上训练每个模型只需要不到1小时的计算时间。我们还可以独自训练VAE和MDN-RNN模型，而不必用尽方法调优超参数。

WORLD MODELS

- 3. 赛车实验

Figure 9. Our agent learning to navigate in `CarRacing-v0`.

WORLD MODELS

- 3. 赛车实验
- 这个实验中，world模型对环境奖励信号开始是一无所知的，只有控制器C可以访问环境中的奖励信息。由于线性控制器模型只有867个参数，所以进化算法如CMA-ES非常适合这个优化任务。

- 我们可以使用z_t，用VAE重构每一时刻看到的信息。

WORLD MODELS

- 3. 赛车实验

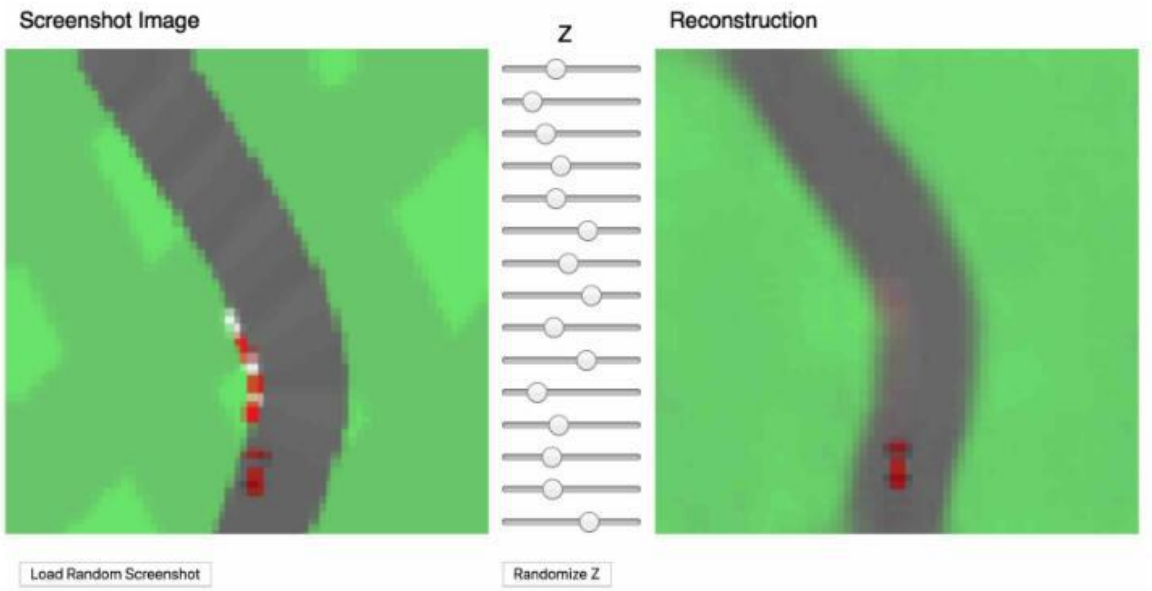


Figure 10. Despite losing details during this lossy compression process, latent vector z captures the essence of each image frame.

在本文给出在线版本链接中，可以加载随机选择的截图，并将其编码为一个小的潜在向量z，用于重构原始截图。也可以尝试使用滑块来调整z向量的值，看看它是如何影响重构的，或者随机调整z来观察可能的屏幕截图的空间。

WORLD MODELS

- 3.赛车实验
- 3.3程序步骤
- 赛车实验步骤如下：
- （1）从一个随机的策略收集1万次滑行的数据
- （2）训练VAE,将图片帧转为潜在向量Z∈R32
- （3）训练MDN-RNN模型
- （4）定义控制器C：at=Wc[zt,ht]+bc
- （5）使用CMA-ES策略求解Wc和bc,并最大化预期的累计奖励。

WORLD MODELS

- 3. 赛车实验
- 3. 3程序步骤(模型参数情况

MODEL	PARAMETER COUNT
VAE	4,348,547
MDN-RNN	422,368
CONTROLLER	867

WORLD MODELS

- 3.赛车实验
- 3.4.Experiment Results
- V Model Only

- 训练一个代理导航驾驶并不难，只要我们对观测有一个很好的表示。先前的工作（Hn-ermann，2017；Bling，2015；Lau，2016）已经表明，通过观察的大量手工设计信息，例如激光雷达信息，角度，位置和速度，人们可以轻松地训练一个小的前馈网络，将这种手工设计的数据输入并输出一个令人满意的导航策略。
- 基于这个原因，我们首先通过限制C只能访问V而不访问M来测试我们的代理。
- 也就是定义我们的代理C：$a_t = W_c z_t + b_c$.(没有隐藏状态)。

WORLD MODELS

- 3. 赛车实验
- 3.4. Experiment Results
- V Model Only

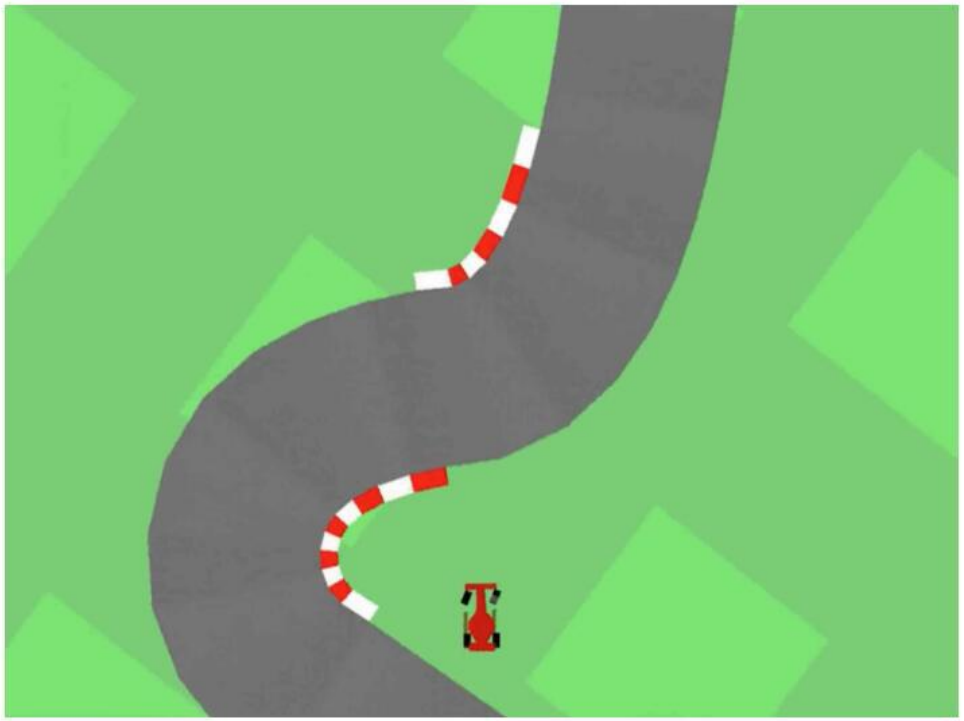


Figure 11. Limiting our controller to see only z_t, but not h_t results in wobbly and unstable driving behaviours.

虽然代理在此设置中仍然能够导航赛道，但我们注意到它会摇摆，在尖角会错过赛道。在100次随机试验中，这个残疾代理的平均得分为632 ±251分，与OpenAIGym上的排名(Klimov, 2016)的其他代理性能一致，和传统的Deep RL方法如A3C (Khan & Elibol, 2016;Jang et al ., 2017)。增加一个隐藏层到C后结果可以改善到788±141，但还没有足够解决这个环境问题。

WORLD MODELS

- 3.4.Experiment Results
- Full World Model(Vand M)
- 我们的V模型获得的表示Z_t，只是某一时刻的表达，并没有很强的预测能力。与此相反，M的训练就是为了预测下一时刻的Z_{t+1}，M预测的Z_{t+1}是由RNN的隐藏状态h_t产生的,因此这个隐藏向量，是我们可以给我们的代理学习特征集的一个很好的候选。两者结合，去预测a_t。

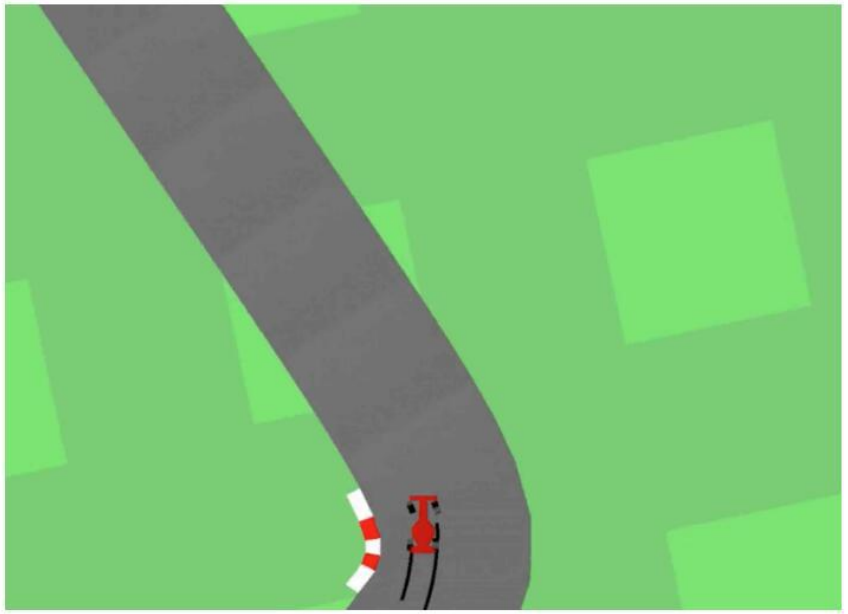


Figure 12. Driving is more stable if we give our controller access to both z_t and h_t.

WORLD MODELS

3.4. Experiment Results

- Full World Model(Vand M)
- 引入了ht信息后大大改善了性能，拐弯处也更准确。此外，我们还看到，在赛车比赛中做出这些快速反射性地驾驶决策时，代理不需要提前计划和推演未来的假想场景。因为这里的ht包含了未来的概率信息，代理可以本能地对RNN进行查询，以指导其行为决策。

METHOD	AVG. SCORE
DQN (PRIEUR, 2017)	343 ± 18
A3C (CONTINUOUS) (JANG ET AL., 2017)	591 ± 45
A3C (DISCRETE) (KHAN & ELIBOL, 2016)	652 ± 10
CEOBILLIONAIRE (GYM LEADERBOARD)	838 ± 11
V MODEL	632 ± 251
V MODEL WITH HIDDEN LAYER	788 ± 141
FULL WORLD MODEL	$\mathbf{906 \pm 21}$

Table 1. CarRacing-v0 scores achieved using various methods.

WORLD MODELS

3.4.Experiment Results

- Full World Model(Vand M)
- 上面表格是随机测试100次的结果。排行榜上100次随机试验最佳的得分是838±11。

- 传统的深度学习方法都需要预处理每一个图片帧（比如利用边缘检(Jangetal.,2017），
- 或者需要叠加一些最近的图片帧到输入中(Khan & Elibol,2016;Jang etal.,2017)。
- 相反，我们的world models接收原始RGB像素图像流，并直接学习时空表示。

WORLD MODELS

- 3.5赛车的梦想
- 既然我们的世界模型能够模拟未来，我们也能够让它自己模拟虚拟赛车场景。给定当前的状态，我们要求模型生成一个z_{t+1}的概率分布，从这个分布中采样一个z_{t+1}，并用它来作为真实的观测，我们把已经训练好的C重新放回到这个由M假想的虚拟环境中去。以下图片是本[illegible]们的world model来幻想一个赛车环境场[illegible]

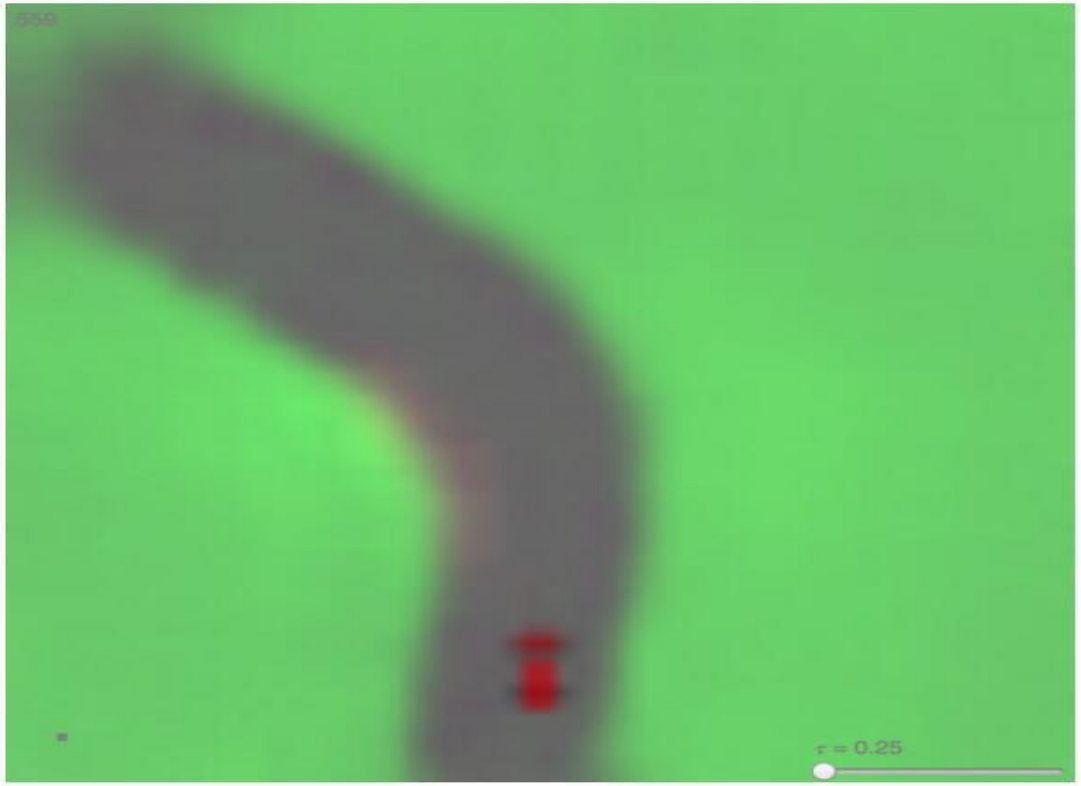


Figure 13. Our agent driving inside of its own dream world. Here,

WORLD MODELS

3.4.Experiment Results

- Full World Model(Vand M)
- 我们的代理在自己假象的世界驾驶，我们将训练好的策略部署到这一个虚假的环境中，这个虚拟环境是由MDN-RNN生成的，并由VAE的解码器渲染生成图片。在本示例中，我们可以推翻代理的行为，只需通过调节 来控制M生成的这个虚拟环境。

- 4. VizDoom Experiment
- VizDoom是一种基于Zoom的AI平台，支持多智能体和竞争环境下测试智能体。
- .....