國 立 中 央 大 學

資訊工程學系

碩士論文

基於自注意力機制產生的無方向性序列編碼器使用同義詞與反義詞資訊調整詞向量

Adjusting Word Embeddings with Synonyms and Antonyms based on Undirected List Encoder Generated from Self-Attention

研 究 生：林冠佑

指導教授：陳弘軒　博士

中 華 民 國 一百零九 年 六 月

國立中央大學圖書館學位論文授權書

填單日期：107 / 7 / 20　　2019.9 版

授權人姓名	林冠佑	學 號	107522053
系所名稱	資工所	學位類別	☑碩士　□博士
論文名稱	基於自注意力機制產生的無方向性序列編碼器使用同義詞與反義詞資訊調整詞向量	指導教授	陳弘軒

學位論文網路公開授權

授權本人撰寫之學位論文全文電子檔：

・在「**國立中央大學圖書館博碩士論文系統**」.

(✓)同意立即網路公開

(　)同意 於西元＿＿＿年＿＿＿月＿＿＿日網路公開

(　)不同意網路公開，原因是：＿＿＿＿＿＿＿＿＿＿

・在**國家圖書館「臺灣博碩士論文知識加值系統」**

(✓)同意立即網路公開

(　)同意 於西元＿＿＿年＿＿＿月＿＿＿日網路公開

(　)不同意網路公開，原因是：＿＿＿＿＿＿＿＿＿＿

依著作權法規定，非專屬、無償授權國立中央大學、台灣聯合大學系統與國家圖書館，不限地域、時間與次數，以文件、錄影帶、錄音帶、光碟、微縮、數位化或其他方式將上列授權標的基於非營利目的進行重製。

學位論文紙本延後公開申請 (紙本學位論文**立即公開者此欄免填**)

本人撰寫之學位論文紙本因以下原因將延後公開

・**延後原因**

(　)已申請專利並檢附證明，專利申請案號：

(　)準備以上列論文投稿期刊

(　)涉國家機密

(　)依法不得提供，請說明：＿＿＿＿＿＿＿＿

・**公開日期**：西元＿＿＿＿＿年＿＿＿＿＿月＿＿＿＿＿日

※繳交教務處註冊組之紙本論文(送繳國家圖書館)若不立即公開，請加填「國家圖書館學位論文延後公開申請書」

研究生簽名：林冠佑　　指導教授簽名：陳弘軒

*本授權書請完整填寫並親筆簽名後，裝訂於論文**封面之次頁**。

國立中央大學碩士班研究生
論 文 指 導 教 授 推 薦 書

資訊工程學系碩士班 學系/研究所 林冠佑 研究生

所提之論文 基於自注意力機制產生的無方向性序列編碼器使用同義詞與反義詞資訊調整詞向量

係由本人指導撰述，同意提付審查。

指導教授 _____ (簽章)

109 年 7 月 14 日

1090706

國立中央大學碩士班研究生
論文口試委員審定書

資訊工程學系碩士班 學系/研究所 林冠佑 研究生

所提之論文 基於自注意力機制產生的無方向性序列編碼器使用同義詞與反義詞資訊調整詞向量

經由委員會審議，認定符合碩士資格標準。

學位考試委員會召集人 _____

委　　員 _____

_____

_____

_____

中　華　民　國　109 年　7 月　7 日

1090706

基於自注意力機制產生的無方向性序列編碼器使用同義詞與反義詞資訊調整詞向量

摘要

自從詞向量被廣泛應用在許多自然語言處理任務，且取得不錯的成果後，學者們開始相信詞向量是可以有效學習到詞義資訊的，並開始研究如何提升詞向量的品質。本論文認為詞向量主要是透過上下文資訊進行學習，沒有利用到人類編撰的字詞關係，如：同/反義字，知識圖譜等，且我們推測詞向量在辨別同義詞與反義詞的能力上仍有進步空間，加入從字典中萃取出的知識應能改善，然而，過去相關的研究僅使用 pairwise 的方法對同義詞與反義詞進行調整，這種方法沒有辦法同時考慮一詞與其所有同義詞和反義詞之間的關係，因此本論文提出了 listwise 的方法來對詞向量進行調整，提升詞向量的品質。

經過實驗，本論文發現採用全局資訊的模型均優於只採用局部資訊的模型，其中學習分配不同注意力在同義詞和反義詞中不同的詞上，再結合這些資訊調整詞向量的自注意力機制更能有效的利用全局資訊，因此本論文選擇使用自注意力機制做為編碼器，在訓練後使用從字典中萃取出的同義詞與反義詞資訊調整詞向量，提升詞向量的品質。為了更多的提升詞向量的品質，本論文嘗試了正規化、殘差連結、多頭式自注意力機制、更深層的神經網路等方法，並設計實驗說明它們對模型的影響。

最後，本論文設計實驗證明經本方法調整後，使用少量文本預訓練的詞向量在同義詞任務中表現可以超越未調整但使用大量文本預訓練的詞向量，並從結果中發現同義詞相較於反義詞在相似度任務上是更有用

的資訊，且同義詞和反義詞資訊並不是越多越好，品質也會影響調整後的結果。

關鍵字：詞向量, 自注意力機制, 同義詞, 反義詞

Adjusting Word Embeddings with Synonyms and Antonyms based on Undirected List Encoder Generated from Self-Attention

Abstract

Since word embedding has become a standard technique working excellently in various natural language processing (NLP) tasks, there has been much research on improving the quality of the word embeddings. We argue that the word embeddings are mainly learned through contextual information but ignore the relationship (e.g., synonyms, antonyms, and knowledge graph) of words compiled by humans. We speculate that including human compiled information may improve the quality of the word embeddings. Unlike previous works, we purpose a listwise method that can consider the relations between a word and its synonyms and antonyms.

Experimental results show that our approach to adjust the word embeddings trained from small corpus yields comparable, sometimes even better, results than the word embeddings trained with a large corpus. Additionally, we show that both the quantity and quality of synonyms and antonyms affect the performance of our work. Finally, we show that models utilizing global information outperform the ones utilizing local information in most cases.

Keywords: Word embeddings, Self-attention, Synonym, Antonym, Post-training, Listwise

目錄

圖目錄

頁次

表目錄

頁次

一、 緒論

1.1 研究動機

在自然語言處理中，若想使用一個向量來表示一個詞，最簡單直觀的方法為 1-of-N Encoding，每個詞的向量維度均為字典長度，而字典中每一個詞對應到一個維度，這種表示法不僅向量維度會隨著文本內容的增加越來越大，且向量本身無法表示詞之間的關係。

為了解決上述問題，科學家們根據分布假說 (Distributional Hypothesis)[1] 提出的方法大致可分為 Count-based 和 Prediction-based[2]，前者利用詞頻統計，使得共同出現越多次的詞，詞向量越接近，著名代表為 GloVe[3]，後者訓練神經網路來預測上下文，並將隱藏層當作詞向量，著名代表為 Word2Vec[4] 以及加入 subword 資訊的 fastText[5]。

自從詞向量被廣泛應用在許多自然語言處理任務，且取得不錯的成果後，學者們開始相信詞向量是可以有效學習到詞義資訊的，並開始研究如何提升詞向量的品質。其中，有學者認為詞向量在訓練時，僅有上下文資訊，因此若加入字典中的知識，如詞性、詞義、詞與詞之間關係等，應能提升詞向量的品質，而他們的方法大致分為兩類：在訓練中加入字典知識 [6]–[9]、在訓練後使用字典知識調整詞向量 [10]–[13]。

受到他們的啓發，我們研究後發現了反義詞也時常擁有相似詞向量的現象，如 1.1所示，我們推測是因雖然兩詞為反義詞，但撇除反義詞關係後，兩詞通常是用來形容同一件事，因此某種程度上來說是相關的詞，且兩詞因詞性相同，導致在句子中可以很容易地被抽換，因此時常擁有

相同的上下文，而根據分布假說，也就是擁有相似上下文的詞會擁有相似的詞義，因此訓練後儘管兩詞為反義詞仍擁有相似的詞向量，所以我們推測詞向量在辨別同義詞與反義詞的能力上仍有進步空間，加入字典中的知識應能改善，並提升詞向量的品質。然而，過去相關的研究僅使用 pairwise 的方法對同義詞與反義詞進行調整，也就是調整時僅使用成對 (pair) 的資料，這種方法通常是將同義詞對拉近，反義詞對拉遠，但這種方法並沒有辦法同時考慮一詞與其所有同義詞和反義詞之間的關係，因此本論文希望能提出一個 listwise 的方法來對詞向量進行調整。

表 1.1: 詞向量在辨別同義詞與反義詞的能力上仍有進步空間

	Top-6 similar words in GloVe					
short	long (.6962)	shorter (.5822)	length (.5626)	longer (.5618)	few (.5042)	only (.4988)
small	large (.7927)	tiny (.7299)	smaller (.7054)	larger (.6562)	few (.6079)	sized (.5986)
cold	warm (.6138)	cool (.5798)	chill (.5726)	temperatures (.5648)	chilly (.5644)	frigid (.5036)
wet	dry (.6799)	damp (.6127)	moist (.5401)	soaking (.5295)	rainy (.5216)	humid (.5215)
easy	easier (.7288)	quick (.6607)	way (.6256)	make (.6243)	difficult (.6236)	simple (.6062)

1.2 研究目標

本論文研究目標為提出 listwise 的方法，使用從字典中萃取出的同義詞與反義詞資訊調整詞向量，由於我們萃取出的同義詞與反義詞資訊通常都是集合 (set) 而不是序列 (sequence)，也就是沒有順序性的資料，因此在模型選擇方面，我們不使用序列到序列模型 (seq2seq)[14] 或是循環神經網路 (Recurrent Neural Network) 及其變形 [15], [16] 等有方向性的模型，選擇無方向性，且會學習分配不同注意力在上下文中不同的詞

上，並結合這些資訊作為輸出調整詞向量的自注意力機制 (Self-Attention Mechanism)[17]。

本論文深入研究 listwise 的方法，設計實驗比較從不同字典中萃取出的資訊對於不同詞向量的影響，驗證了正規化、殘差連結、自注意力機制的全局資訊對於模型的影響，也解釋了目標函式設計對於模型的意義等較為細節的內容。

1.3 研究貢獻

本論文提出了 listwise 的方法，使用從字典中萃取出的同義詞與反義詞資訊調整詞向量，以提升詞向量的品質。本方法相較於 pairwise 的方法，可以同時考慮一詞與其所有同義詞和反義詞之間的關係；相較於在訓練中加入字典知識的方法，除了可以節省時間及資源，在擴充字典知識方面也方便了許多，且可以應用在所有詞向量上。

本方法只要能取得可視為同義詞集合的資訊，便可以應用於所有用到嵌入 (embedding) 或是向量 (vector) 來表示實體的應用，例如在電商平台可以使用商品類型資訊來調整表示商品的向量，或是用使用者資訊來調整表示使用者的向量等。

1.4 論文架構

本論文共分為六個章節，架構如下:

第一章、說明本篇論文之研究動機、研究目標、研究貢獻。

第二章、介紹與本論文相關的研究。

第三章、說明本論文模型架構、設計原因以及所用技術。

第四章、介紹本論文實驗所用資料。

第五章、說明本論文實驗設計與實驗結果探討。

第六章、本論文之結論與未來展望。

一、緒論

二、 相關研究

2.1 Retrofitting Word Vectors to Semantic Lexicons

使用從字典中萃取出的同義詞資訊調整預訓練好的詞向量的方法，最早由該論文 [10] 提出，該論文使用預訓練好的詞向量 $\hat{Q} = (\hat{q}_1, \ldots, \hat{q}_n)$ 初始化一個新的向量空間 $Q = (q_1, \ldots, q_n)$，並設計目標函式最小化新的向量空間 Q 中同義詞對 (i, j) 之間的距離，其詳細定義如下：

$$\Psi(Q) = \sum_{i=1}^{n} \left[\alpha_i \left\| q_i - \hat{q}_i \right\|^2 + \sum_{(i,j) \in E} \beta_{ij} \left\| q_i - q_j \right\|^2 \right] \tag{2.1}$$

其中 E 為從字典中萃取出的同義詞對資料集，α、β 為控制權重用的超參數，前者控制新的向量空間與原向量空間差異的程度，後者控制同義詞在新的空間向量中向彼此接近的程度。

2.2 Counter-fitting Word Vectors to Linguistic Constraints

該論文 [11] 在調整預訓練好的詞向量時，使用預訓練好的詞向量 $V = (\mathbf{v}_1, \ldots, \mathbf{v}_n)$ 初始化一個新的向量空間 $V' = (\mathbf{v}'_1, \ldots, \mathbf{v}'_n)$，設計目標函式在新的向量空間 V' 中最小化同義詞對之間的距離和最大化反義詞

對之間的距離，同時因認為原向量空間中有著詞與詞之間的關係的資訊，因此應維持在原向量空間中相鄰的詞調整前後的距離，其詳細定義如下：

$$\mathrm{AR}\left(V'\right) = \sum_{(u,w)\in A} \tau\left(\delta - d\left(\mathbf{v}'_u, \mathbf{v}'_w\right)\right) \tag{2.2}$$

AR(Antonym Repel) 將反義詞的詞向量推遠，其中 $d\left(v_i, v_j\right) = 1 - \cos\left(v_i, v_j\right)$ 使用餘弦相似度計算距離，$\tau(x) \triangleq \max(0, x)$ 讓距離已足夠遠的反義詞不用再更遠，δ 為反義詞距離的理想值，在該論文中 $\delta = 1.0$，A 為從字典中萃取出的反義詞資料集。

$$\mathrm{SA}\left(V'\right) = \sum_{(u,w)\in S} \tau\left(d\left(\mathbf{v}'_u, \mathbf{v}'_w\right) - \gamma\right) \tag{2.3}$$

SA(Synonym Attract) 將同義詞的詞向量拉近，其中 $\tau(x) \triangleq \max(0, x)$ 讓距離已足夠近的同義詞不用再更近，γ 為同義詞距離的理想值，在該論文中 $\gamma = 0$，S 為從字典中萃取出的同義詞資料集。

$$\mathrm{VSP}\left(V, V'\right) = \sum_{i=1}^{N} \sum_{j\in N(i)} \tau\left(d\left(\mathbf{v}'_i, \mathbf{v}'_j\right) - d\left(\mathbf{v}_i, \mathbf{v}_j\right)\right) \tag{2.4}$$

VSP(Vector Space Preservation) 保留原空間的資訊，其中 $\tau(x) \triangleq \max(0, x)$ 讓調整後距離變近的相鄰詞不用再調整，只調整距離變遠的相鄰詞，$N(i)$ 為在原向量空間中與 $\mathbf{v}_i$ 相鄰的詞向量集合，而為求計算效率該論文將相鄰定義為與 $\mathbf{v}_i$ 距離 ρ 以內，在該論文中 $\rho = 0.2$。

$$C\left(V, V'\right) = k_1\mathrm{AR}\left(V'\right) + k_2\mathrm{SA}\left(V'\right) + k_3\mathrm{VSP}\left(V, V'\right) \tag{2.5}$$

最後，將三者結合即為損失函數，其中 k_1、k_2、k_3 為權重參數。

2.3 Adjusting Word Embeddings with Semantic Intensity Orders

該論文 [12] 在調整預訓練好的詞向量時分為兩個步驟, 在使用預訓練好的詞向量 $V^0 = \left(v_1^0, \ldots, v_n^0\right)$ 初始化一個新的向量空間 $V = (v_1, \ldots, v_n)$, 並加入從字典中萃取出的同義詞和反義詞資訊及保留原向量空間中的資訊後，額外加入了詞義程度的資訊，認為在調整同義詞及反義詞距離時應依詞義程度進行調整，其詳細定義如下:

$$AF(V) = \sum_{(u,w)\in A} \tau\left(\cos\left(v_u, v_w\right)\right) \tag{2.6}$$

AF 調整反義詞之間的距離，其中 $\tau(x) \triangleq \max(0, x)$ 讓距離已足夠遠的反義詞不用再更遠，A 為從字典中萃取出的反義詞集合。

$$SC(V) = \sum_{(u,w)\in S} \tau\left(1 - \cos\left(v_u, v_w\right)\right) \tag{2.7}$$

SC 調整同義詞之間的距離，其中 $\tau(x) \triangleq \max(0, x)$ 讓距離已足夠近的同義詞不用再更近，S 為從字典中萃取出的同義詞集合。

$$KN(V, V^0) = \sum_{i=1}^{N} \sum_{j\in N(i)} \tau\left(\cos\left(v_i, v_j\right) - \cos\left(v_i^0, v_j^0\right)\right) \tag{2.8}$$

KN 維持在原向量空間中相鄰的詞調整前後的距離，$\tau(x) \triangleq \max(0, x)$ 讓調整後距離變近的相鄰詞不用再調整，只調整距離變遠的相鄰詞，$N(i)$ 為在原向量空間中與 v_i^0 相鄰的詞集合，而該論文對相鄰的定義為與 v_i^0 餘弦相似度大於 0.8 即為相鄰。

$$C\left(V, V^0\right) = AF(V) + SC(V) + KN\left(V, V^0\right) \tag{2.9}$$

將以上三者結合即為第一步驟的損失函數。

$$SO(V) = \sum_{(u,w)\in E} \tau\left(1 - \cos\left(v_u, v_w\right)\right) \tag{2.10}$$

SO 使得所有詞與詞義程度相同的詞更為接近,其中 $\tau(x) \triangleq \max(0, x)$ 讓距離已足夠接近的詞對不用再更接近，E 為透過 k-means++ 分群演算法取得擁有相同詞義程度的詞對資料集。

$$AO(V) = \sum_{(w,a)\in A} \sum_{s\in \mathrm{Str}(w)} \tau\left\{\cos\left(v_s, v_a\right) - \cos\left(v_w, v_a\right)\right\} \tag{2.11}$$

AO 使得所有詞與反義詞中詞義程度較强的詞更為遙遠，其中 $\tau(x) \triangleq \max(0, x)$ 讓距離已足夠遙遠的詞對不用再更遙遠，A 為從字典中萃取出的反義詞集合，$\mathrm{Str}(w)$ 為透過分群演算法取得詞義程度較 w 强的詞對資料集。

2.4 Extrofitting: Enriching Word Representation and its Vector Space with Semantic Lexicons

該論文 [13] 不同於上述三者，在調整預訓練好的詞向量時分為三個步驟，首先將預訓練好的詞向量 $q_i = (e_{i1}, e_{i2}, \cdots, e_{iD})$ 擴展數個維度，並於擴展的每個維度填入其與其同義詞的平均值 $\bar{r}_i = \sum_{s\in L} r_s/N$，其中 $r_i = \mathrm{mean}\left(e_{i1}, e_{i2}, \cdots, e_{iD}\right)$ 為 q_i 的平均值，s 為同義詞對 (pair)，L 為 s 的集合，而 N 為 L 的大小，接著再使用從字典中萃取出的同義詞資訊將詞標籤，最後使用線性判別分析 (Linear Discriminant Analysis) 將擴展後的詞向量降維至原本的維度得到新的詞向量 $\bar{Q} = LDA(Q \oplus \bar{r}, l)$，其中

l 為標籤資訊。

2.5 相關研究性質

最後，我們整理了相關研究的特點，以及與本論文的差異，結果如表 2.1。其中包含的特點有：是否使用同義詞調整詞向量、是否使用反義詞調整詞向量、是否使用同反義詞外的其他知識調整詞向量、是否能便利的擴充知識、是否屬於在訓練後加入知識 (post-training) 的方法、是否屬於 listwise 的方法。

表 2.1: 相關研究性質

	同義詞	反義詞	其他知識	可擴充知識	post-training	listwise
Improving[6]	√	√	√			
Retrofitting[10]	√			√	√	
Counter-fitting[11]	√	√		√	√	
Adjusting[12]	√	√	√	√	√	
Extrofitting[13]	√				√	√
Our Work	√	√		√	√	√

三、 模型及方法

3.1 模型架構

本論文將字典中的同義詞和反義詞資訊加入預訓練詞向量的字典 W，若一詞 $w_i \in W$，且有同義詞集合 $S = \{s_1, \ldots, s_n\}$ 和反義詞集合 $A = \{a_1, \ldots, a_m\}$，則我們會將 S 和 A 接在 w_i 之後，處理成序列做為模型輸入，經過多層無方向性的序列編碼器和殘差連接，輸出含有更豐富詞義資訊的詞向量，如圖 3.1所示。

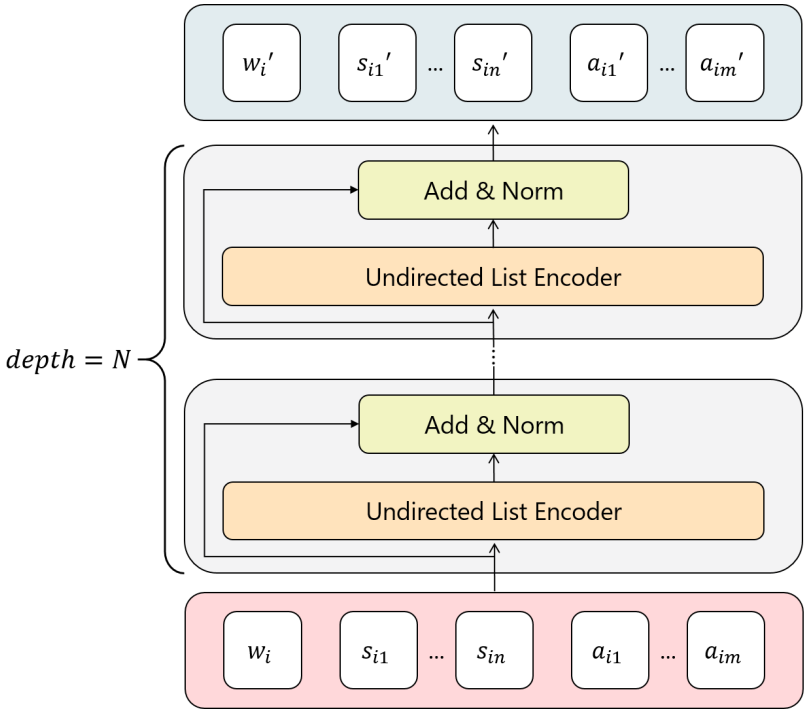


圖 3.1: 模型架構

3.2 無方向性的序列編碼器

本論文希望模型的每一個輸出都有學習到整個輸入序列所有的同義詞和反義詞資訊，但是由於輸入的同義詞集合與反義詞集合並沒有順序性，因此本論文認為輸入序列的順序不應影響輸出結果，故不使用自然語言處理中常用的循環神經網路及其變形，改用無方向性的序列模型，也就是自注意力機制做為本論文模型的編碼器。

3.2.1 自注意力機制 (Self-Attention Mechanism)

[17] 提出了自注意力機制，其精神為產生每一個輸出時，都會分配注意力在輸入序列所有的元素上，然後合併所有注意到的資訊做為自己的輸出，這種模型不僅和循環神經網路一樣可以處理序列型資料，還可以平行化運算，更重要的是輸入序列的順序不影響輸出結果，以上優點都非常符合本論文的需求，以下將簡單介紹其原理。

自注意力機制將輸入序列進行線性轉換後得到 query、key、value，並計算每一個元素的 query 與其他元素的 key 之間的相似度分數，也就是要分配多少注意力在其他元素上，並使用該分數做為權重，對其他元素的 value 做加權和得到輸出，詳細公式如下：

$$\text{Attention}\ (Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.1}$$

其中 Q、K、V 分別為 query、key、value 所組成的矩陣，d_k 為 key 的維度，該篇論文中將內積結果使用 softmax 函數進行歸一化計算權重，並除以 d_k 以防止內積結果過大。

除了上述被稱為點積式注意力 (Scaled Dot-Product Attention) 的方法外，[17] 還提出了多頭式注意力機制 (Multi-Head Attention)，將 query、key、value 進行線性轉換 h 次，分別輸入點積式注意力中訓練，並將結果拼接起來得到輸出，這種方法可以結合從不同維度、不同子空間中學

習到的資訊，是單頭式所做不到的，詳細公式如下：

$$\begin{aligned} \text{MultiHead}\ (Q,K,V) &= \ \text{Concat}(\text{head}\ _1,\ldots,\ \text{head}\ _\text{h})\,W^O \\ \text{where head}\ _i &= \ \text{Attention}\ \left(QW_i^Q, KW_i^K, VW_i^V\right) \end{aligned} \tag{3.2}$$

其中 W_i^Q、W_i^K、W_i^V、W^O 皆為線性轉換矩陣。

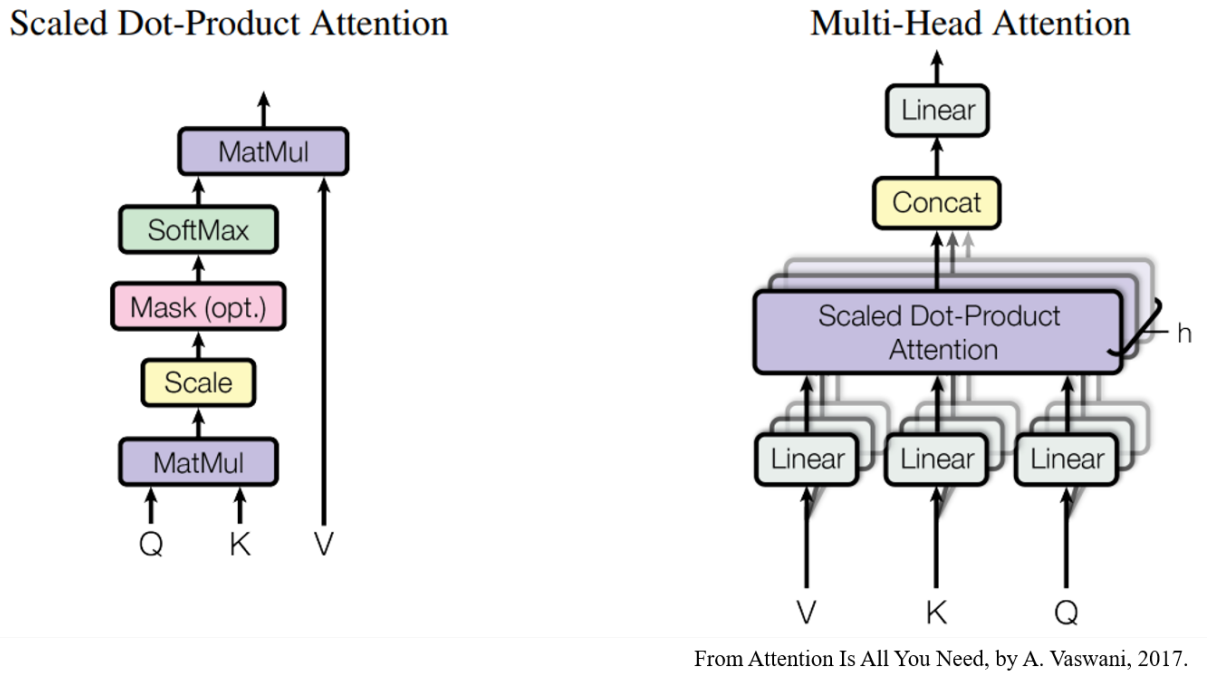


圖 3.2: (左) 點積式注意力 (右) 多頭式注意力

3.2.2 殘差連結 (Residual Connections)

理論上，多層的神經網路學習能力應較少層的佳，但在堆疊多層的神經網路時，容易產生梯度消失或梯度爆炸等問題，因此我們同 [17] 使用了殘差連結的做法，使得每一個編碼器 (Encoder) 的輸出為 $\text{LayerNorm}(x + \text{Encoder}(x))$。

3.3 損失函數

本論文所使用之損失函數如下：

$$l_1 = \frac{\sum_{s \in S} \left(1 - \cos\left(w_i', s'\right)\right) + \sum_{a \in A} \cos\left(w_i', a'\right)}{n + m} \tag{3.3}$$

$$l_2 = \frac{\operatorname{mse}\left(w_i, w_i'\right) + \sum_{s \in S} \operatorname{mse}\left(s, s'\right) + \sum_{a \in A} \operatorname{mse}\left(a, a'\right)}{1 + n + m} \tag{3.4}$$

$$\text{Loss } = \alpha l_1 + (1 - \alpha)\, l_2 \tag{3.5}$$

其中 w_i 為調整前的詞向量，w_i' 為調整後的向量，S 為 w_i 的同義詞集合，n 為 S 的大小，A 為 w_i 的反義詞集合，m 為 A 的大小，α 為控制調整程度的超參數，cos 為餘弦相似度 (cosine similarity)，mse 為均方差 (mean-square error)。

在損失函數 l_1 中，我們設計 $\left(1 - \cos\left(w_i', s'\right)\right)$，其目的為最大化 w_i' 與其同義詞在調整後的餘弦相似度，靈感來源為餘弦距離公式，若兩詞餘弦相似度越高，則餘弦距離越小。同時，設計 l_1 中 $\cos\left(w_i', a'\right)$，其目的為最小化 w_i' 與其反義詞在調整後的餘弦相似度。

在損失函數 l_2 中，我們透過均方差維持詞向量調整前後的差異，以保留原向量空間中的資訊。值得一提的是，我們曾嘗試使用餘弦相似度來維持詞向量在調整前後差異，但我們發現餘弦相似度高僅代表調整前後夾角相近，但實際上在向量空間中的位置可能差異很大，因此在這邊使用均方差可能更為合適。

四、 實驗資料集

4.1 預訓練詞向量

GloVe[3] 是一種基於全局詞頻統計訓練出來的詞向量，本論文使用的 GloVe_{6B} 是官方以 Wikipedia、Gigaword 預訓練的。GloVe_{42B} 則是官方以 Common Crawl 預訓練的，兩者皆可以從官方網站 [1] 取得。

Word2Vec[4] 是一種使用滑動窗口取得局部上下文資訊訓練而得的詞向量，本論文使用的 Word2Vec 是 [18] 以 Wikipedia 為語料，用 skip-gram 模型預訓練的，可從 GitHub[2] 取得。

fastText[5] 是一種在 Word2Vec 的方法上加入 subword 資訊，使得詞向量含義更加豐富的一種方法，本論文使用的 fastText 是官方以 Wikipedia、UMBC webbase corpus、statmt.org news dataset 預訓練的，可從官方網站 [3] 取得。

4.2 同反義詞字典

PPDB[19] 是一個有 16 種語言共數百萬組釋義 (paraphrases) 的資料庫，根據 PPDB2.0 中新加入的特徵—關係，若兩個詞的關係為 Equivalence 則視為同義詞，若兩個詞的關係為 Exclusion 則視為反義詞。

[1] https://nlp.stanford.edu/projects/glove/
[2] https://wikipedia2vec.github.io/wikipedia2vec/pretrained/
[3] https://fasttext.cc/docs/en/english-vectors.html

此外，我們借用 [10] 的方法，若兩個不同的詞在翻譯成另一個語言後為同一個詞，則將他們視為同義詞。透過以上兩種方法，我們萃取出只有同義詞資訊的字典 PPDB_{syn}，和加入了反義詞資訊的字典 $\text{PPDB}_{\text{syn+ant}}$，但我們發現 $\text{PPDB}_{\text{syn+ant}}$ 中同義詞與反義詞的例相當懸殊，反義詞的數量僅占 1%，因此我們使用 [20] 從 PPDB 萃取出的反義詞資料集 [4]，加入 $\text{PPDB}_{\text{syn+ant}}$ 產生新的字典 $\text{PPDB}_{\text{syn+ant+}}$，在 $\text{PPDB}_{\text{syn+ant+}}$ 中反義詞占 44%。

WordNet[21] 是一個以英文為主的語義網路，它將同義詞組成一個同義詞集合 (synset)，並記錄同義詞集合之間的關係。我們使用所有的同義詞集合當作同義詞字典 $\text{WordNet}_{\text{syn}}$，再加入關係為 antonymy 的反義詞得到字典 $\text{WordNet}_{\text{syn+ant}}$。

FrameNet[22] 由超過 20 萬筆標記過的句子及對應的超過 1200 種語義框架組成，我們借用 [10] 的方法，認為連結到相同語義框架的詞擁有相近的語義，得到同義詞字典 $\text{FrameNet}_{\text{syn}}$。

4.3 相似度任務

本論文使用相似度任務來評估模型的表現，輸入一系列相似詞後，以模型輸出的相似度與人工標註的相似度之間的斯皮爾曼等級相關係數作為評量標準。本論文使用的 **MEN-3k[23]** 含有 3000 組相似詞及 0～50 分的相似度分數，**SL-999[24]** 含有 999 組相似詞及 0～10 分的相似度分數，**WS-353[25]** 含有 353 組相似詞及 0～10 分的相似度分數，**RG-65[26]** 含有 65 組相似詞及 0～4 分的相似度分數。

[4] https://github.com/srajana/learning-antonyms-ppdb

五、 實驗結果

5.1 使用從不同字典中萃取出的知識調整詞向量

本實驗使用從不同字典中萃取出的知識對 GloVe_{6B} 進行調整，並比較其在各個相似度任務中的表現，結果如表 5.1、表 5.2、表 5.3。

從表 5.1可以發現，調整前的 $\text{GloVe}_{\text{42B}}$ 的結果大多優於調整前的 GloVe_{6B}，因此我們可以相信使用更大量的語料進行訓練，雖然更加耗費時間與資源，但詞向量所學習到的語義更為精確，然而本論文使用從字典中萃取出的知識對 GloVe_{6B} 進行調整後，其結果幾乎都可以接近甚至超越 $\text{GloVe}_{\text{42B}}$。

表 5.1: 使用從不同字典中萃取出的知識調整詞向量 (一)

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	0.7693
$\text{GloVe}_{\text{42B}}$	0.7435	0.3725	**0.6376**	**0.8172**
PPDB_{syn}	0.7592 ± 0.0009	0.3741 ± 0.0011	0.6328 ± 0.0043	0.7530 ± 0.0057
$\text{PPDB}_{\text{syn+ant}}$	0.7596 ± 0.0006	0.3726 ± 0.0011	0.6282 ± 0.0045	0.7574 ± 0.0040
$\text{WordNet}_{\text{syn}}$	0.7608 ± 0.0004	0.3663 ± 0.0010	0.6321 ± 0.0055	0.7684 ± 0.0103
$\text{WordNet}_{\text{syn+ant}}$	$\mathbf{0.7612 \pm 0.0031}$	$\mathbf{0.3751 \pm 0.0020}$	0.6334 ± 0.0084	0.7677 ± 0.0063
$\text{FrameNet}_{\text{syn}}$	0.7582 ± 0.0035	0.3681 ± 0.0077	0.6104 ± 0.0131	0.7663 ± 0.0095

從表 5.1還可以發現 $\text{WordNet}_{\text{syn+ant}}$ 大多優於 $\text{WordNet}_{\text{syn}}$，且 $\text{PPDB}_{\text{syn+ant}}$ 大多優於 PPDB_{syn}，因此我們認為雖然目前加入的反義詞資訊不多，但反義詞對於調整詞向量是有幫助的資訊，若能加入更多的反義詞資訊結果應該可以有更顯著的提升，於是便使用了 [20] 的方法，從 PPDB 萃取

了大量的反義詞加入訓練，結果如表 5.2，得到的實驗結果不如預期，反而是 $\text{PPDB}_{\text{syn+ant}}$ 大多優於 $\text{PPDB}_{\text{syn+ant+}}$，這讓我們產生以下推測：同義詞相較於反義詞在相似度任務上是更有用的資訊，且同義詞和反義詞資訊並不是越多越好，品質也會影響調整後的結果。

表 5.2: 使用從不同字典中萃取出的知識調整詞向量 (二)

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	0.7693
$\text{PPDB}_{\text{syn+ant}}$	$\mathbf{0.7596 \pm 0.0006}$	$\mathbf{0.3726 \pm 0.0011}$	$\mathbf{0.6282 \pm 0.0045}$	0.7574 ± 0.0040
$\text{PPDB}_{\text{syn+ant+}}$	0.7511 ± 0.0010	0.3718 ± 0.0023	0.6082 ± 0.0041	$\mathbf{0.7620 \pm 0.0083}$

承上述推測，這其實和我們從相關研究中觀察到的並不相同，相關研究中顯示調整後詞向量的表現與被調整的詞的數量成正相關，也就是說同義詞和反義詞的資訊越多，效果越好。因此，我們嘗試將所有同義詞字典合併成只有同義詞的 $\text{Lexicons}_{\text{syn}}$、含有少量反義詞的 $\text{Lexicons}_{\text{syn+ant}}$、含有大量反義詞的 $\text{Lexicons}_{\text{syn+ant+}}$，並使用他們調整詞向量，實驗結果如表 5.3，從實驗結果可以發現 $\text{Lexicons}_{\text{syn}}$ 和 $\text{Lexicons}_{\text{syn+ant}}$ 如推測的雖然不錯但幾乎無法超越 $\text{WordNet}_{\text{syn}}$ 和 $\text{WordNet}_{\text{syn+ant}}$，而 $\text{Lexicons}_{\text{syn+ant+}}$ 也如推測的無法超過前面兩者。

表 5.3: 使用從不同字典中萃取出的知識調整詞向量 (三)

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	0.7693
$\text{WordNet}_{\text{syn+ant}}$	$\mathbf{0.7612 \pm 0.0031}$	$\mathbf{0.3751 \pm 0.0020}$	$\mathbf{0.6334 \pm 0.0084}$	$\mathbf{0.7677 \pm 0.0063}$
$\text{Lexicons}_{\text{syn}}$	0.7611 ± 0.0011	0.3781 ± 0.0004	0.6247 ± 0.0008	0.7548 ± 0.0042
$\text{Lexicons}_{\text{syn+ant}}$	0.7610 ± 0.0005	0.3769 ± 0.3769	0.6219 ± 0.0030	0.7570 ± 0.0055
$\text{Lexicons}_{\text{syn+ant+}}$	0.7572 ± 0.0006	0.3755 ± 0.0005	0.6163 ± 0.0034	0.7629 ± 0.0046

5.2 使用不同模型做爲編碼器

本實驗使用不同模型做為編碼器，使用 $\text{WordNet}_{\text{syn+ant}}$ 對 GloVe_{6B} 進行調整，並比較其在各個相似度任務中的表現，結果如表 5.4、表 5.5、表 5.6、表 5.7。

實驗中我們使用卷積核 (kernel) 大小為 3 的一維卷積模型 Conv 代表只採用局部資訊的模型，線性轉換層 Linear 代表採用全局資訊的簡單模型，以及本論文使用的自注意力機制 Attn，結果如表 5.4，從結果中可以發現 Attn 均優於 Linear，且兩者均優於 Conv，我們認為這說明了採用全局資訊的方法較只採用局部資訊的方法佳，且分配不同注意力在上下文中不同的詞上，再結合這些資訊調整詞向量更能有效的利用全局資訊。另外，採用局部資訊的 Conv 在調整上仍是有效果的，我們認為是因為輸入序列是將反義詞集合接在同義詞集合後，而在反義詞集合內的每個詞均互為同義詞，因此我們可將輸入序列視為是將兩個不同的同義詞集合連接在一起，如此一來，使用局部資訊來調整詞向量時，由於局部資訊內的詞均互為同義詞，因此我們可將其視為使用長度較短的同義詞序列來調整詞向量。

表 5.4: 使用不同模型做為編碼器

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	0.7693
Attn	$\mathbf{0.7612 \pm 0.0031}$	$\mathbf{0.3751 \pm 0.0020}$	$\mathbf{0.6334 \pm 0.0084}$	$\mathbf{0.7769 \pm 0.0190}$
Conv	0.7611 ± 0.0009	0.3685 ± 0.0053	0.6280 ± 0.0087	0.7685 ± 0.0191
Linear	0.7611 ± 0.0011	0.3745 ± 0.0054	0.6290 ± 0.0048	0.7616 ± 0.0186

為了更多的提升調整後詞向量的品質，本論文實驗了多頭式自注意力機制 $\text{Attn}_{\text{2heads}}$、$\text{Attn}_{\text{4heads}}$ 是否能從從不同維度、不同子空間中學習到更多的資訊，結果如表 5.5，從結果中可以發現其效果並不顯著。

表 5.5: 多頭式自注意力機制

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	0.7693
Attn	0.7612 ± 0.0031	0.3751 ± 0.0020	$\mathbf{0.6334 \pm 0.0084}$	$\mathbf{0.7769 \pm 0.0190}$
$\text{Attn}_{2\text{heads}}$	0.7606 ± 0.0012	$\mathbf{0.3762 \pm 0.0027}$	0.6293 ± 0.0074	0.7594 ± 0.0079
$\text{Attn}_{4\text{heads}}$	$\mathbf{0.7613 \pm 0.0019}$	0.3737 ± 0.0022	0.6290 ± 0.0034	0.7655 ± 0.0031

除此之外，一般認為深層的神經網路學習效果會較淺層的好，因此本論文嘗試了較多層的 Attn $\times 2$、Conv $\times 2$、Linear $\times 2$，結果如表 5.6，從實驗結果中可以發現他們的效果也並不顯著。其中，Attn $\times 2$ 由於梯度爆炸的問題，本論文嘗試了正規化、設計殘差連結、調整參數等方法仍無法解決，故無法回報實驗結果，但我們認為若是能解決該問題，Attn $\times 2$ 的結果應該會是最好的。

表 5.6: 更深層的神經網路

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	0.7693
Attn	0.7612 ± 0.0031	0.3751 ± 0.0020	$\mathbf{0.6334 \pm 0.0084}$	$\mathbf{0.7769 \pm 0.0190}$
Conv	0.7611 ± 0.0009	0.3685 ± 0.0053	0.6280 ± 0.0087	0.7685 ± 0.0191
Conv $\times 2$	0.7580 ± 0.0031	0.3617 ± 0.0038	0.6312 ± 0.0029	0.7754 ± 0.0095
Linear	0.7611 ± 0.0011	0.3745 ± 0.0054	0.6290 ± 0.0048	0.7616 ± 0.0186
Linear $\times 2$	$\mathbf{0.7624 \pm 0.0020}$	$\mathbf{0.3752 \pm 0.0042}$	0.6323 ± 0.0019	0.7580 ± 0.0116

出於上述問題，本論文認為既然無法解決梯度爆炸，正規化和殘差連結是否就沒有存在的必要? 因此實驗了將正規化和殘差連結移除的 Attn $-$ Res、Conv $-$ Res、Linear $-$ Res，結果如表 5.7，從實驗結果中可以發現正規化和殘差連結對於模型來說還是必要的。

表 5.7: 殘差連結

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	**0.7486**	**0.3692**	**0.6086**	**0.7693**
$\text{Attn} - \text{Res}$	0.1142 ± 0.0137	0.0263 ± 0.0228	0.0458 ± 0.0301	0.2037 ± 0.0274
$\text{Conv} - \text{Res}$	0.0254 ± 0.0170	0.0201 ± 0.0083	0.0185 ± 0.0168	0.0415 ± 0.0310
$\text{Linear} - \text{Res}$	0.2771 ± 0.0178	0.0141 ± 0.0131	0.2876 ± 0.0150	0.2838 ± 0.0411

5.3 從不同字典中萃取出的知識調整不同的預訓練詞向量

本實驗使用從不同字典中萃取出的知識對不同的預訓練詞向量進行調整，並比較其在各個相似度任務中的表現，結果如表 5.8、表 5.9、表 5.10。

從實驗結果可以發現，本論文的方法使用在 Glove 上大多有效，使用在 Word2Vec 上則偶爾有效，但使用在 fastText 上均無效果，和相關研究顯示的結果一致，我們推測和 Count-based 與 Prediction-based 的差異有關，但仍須研究證實。

表 5.8: 使用從不同字典中萃取出的知識調整 GloVe

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	**0.7693**
$\text{PPDB}_{\text{syn+ant}}$	0.7596 ± 0.0006	0.3726 ± 0.0011	0.6282 ± 0.0045	0.7574 ± 0.0040
$\text{WordNet}_{\text{syn+ant}}$	$\mathbf{0.7612 \pm 0.0031}$	$\mathbf{0.3751 \pm 0.0020}$	$\mathbf{0.6334 \pm 0.0084}$	0.7677 ± 0.0063
$\text{FrameNet}_{\text{syn}}$	0.7582 ± 0.0035	0.3681 ± 0.0077	0.6104 ± 0.0131	0.7663 ± 0.0095

表 5.9: 使用從不同字典中萃取出的知識調整 Word2Vec

	MEN-3k	SL-999	WS-353	RG-65
Word2Vec	0.7516	0.3821	0.6960	0.7846
$\text{PPDB}_{\text{syn+ant}}$	$\mathbf{0.7544 \pm 0.0007}$	0.3941 ± 0.0015	$\mathbf{0.7106 \pm 0.0017}$	0.7856 ± 0.0046
$\text{WordNet}_{\text{syn+ant}}$	0.7379 ± 0.0013	$\mathbf{0.3965 \pm 0.0018}$	0.7079 ± 0.0039	$\mathbf{0.7905 \pm 0.0041}$
$\text{FrameNet}_{\text{syn}}$	0.6762 ± 0.0040	0.3811 ± 0.0101	0.6468 ± 0.0143	0.7099 ± 0.0233

表 5.10: 使用從不同字典中萃取出的知識調整 fastText

	MEN-3k	SL-999	WS-353	RG-65
fastText	**0.8032**	**0.4430**	**0.6992**	**0.8843**
$\text{PPDB}_{\text{syn+ant}}$	0.6907 ± 0.0036	0.2676 ± 0.0011	0.6325 ± 0.0019	0.8027 ± 0.0141
$\text{WordNet}_{\text{syn+ant}}$	0.5341 ± 0.0089	0.1554 ± 0.0035	0.4760 ± 0.0083	0.3814 ± 0.0159
$\text{FrameNet}_{\text{syn}}$	0.3368 ± 0.0020	0.2033 ± 0.0053	0.4037 ± 0.0112	0.4596 ± 0.0381

5.4 輸入序列長度

理論上，本論文提出之模型輸入的序列長度應越長越好，理想值是與同義詞集合和反義詞集合中最長的兩筆資料總和等長，但實際上我們發現若依此設定進行實驗，首先遇到的問題是顯存裝不下如此龐大的神經網路，再來是就算克服了硬體限制，過久的訓練時間已失去本方法的優勢，因此本實驗的目的是找出折衷後輸入序列的最佳長度，實驗的長度為各同義詞字典資料長度的四分位數 (Q_1,Q_2,Q_3)，結果如表 5.11。由實驗結果我們可以發現，大多情況下比較大的四分位數 (Q_3) 能得到較好的結果，因此本論文最後選用其作為模型輸入序列的長度。

表 5.11: 輸入序列長度之研究，從結果我們可以發現大多情況下比較大的四分位數 (Q_3) 能得到較好的結果。

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	0.7693
PPDB_{Q1}	0.7596 ± 0.0015	$\mathbf{0.3756 \pm 0.0037}$	0.6311 ± 0.0033	0.7543 ± 0.0034
PPDB_{Q2}	0.7597 ± 0.0017	0.3715 ± 0.0013	0.6323 ± 0.0042	0.7543 ± 0.0082
PPDB_{Q3}	$\mathbf{0.7598 \pm 0.0005}$	0.3726 ± 0.0003	$\mathbf{0.6333 \pm 0.0011}$	$\mathbf{0.7547 \pm 0.0035}$
WordNet_{Q1}	0.7576 ± 0.0016	0.3747 ± 0.0026	$\mathbf{0.6316 \pm 0.0069}$	0.7661 ± 0.0070
WordNet_{Q2}	0.7592 ± 0.0015	$\mathbf{0.3748 \pm 0.0027}$	0.6255 ± 0.0016	0.7632 ± 0.0181
WordNet_{Q3}	$\mathbf{0.7618 \pm 0.0017}$	0.3721 ± 0.0030	0.6282 ± 0.0048	$\mathbf{0.7688 \pm 0.0117}$
FrameNet_{Q1}	0.7434 ± 0.0017	0.3722 ± 0.0098	$\mathbf{0.6174 \pm 0.0118}$	0.7548 ± 0.0161
FrameNet_{Q2}	0.7549 ± 0.0036	0.3675 ± 0.0047	0.6171 ± 0.0136	$\mathbf{0.7720 \pm 0.0087}$
FrameNet_{Q3}	$\mathbf{0.7608 \pm 0.0032}$	$\mathbf{0.3826 \pm 0.0070}$	0.6074 ± 0.0097	0.7718 ± 0.0135

5.5 詞向量訓練時間

本實驗回報詞向量訓練文本大小與其所需的訓練時間，以此推估其他實驗中所使用的開源詞向量預訓練所需的時間，並與本論文提出的方法所需的訓練時間比較，結果如圖 5.1。

本實驗使用 text8[1] 作為語料訓練 GloVe 詞向量，使用的是 6 核心、基礎頻率 1.70GHz 的處理器 [2]，從圖 5.1中可以看出，在訓練的一開始，平均每 1M 個 token 需要 5 秒的訓練時間，但隨著文本增加，平均每 1M 個 token 所需要的訓練時間也隨之增加，且成長得非常迅速，因此我們可以推測使用我們的機器訓練 6B 個 token 的文本至少需要 8.3 小時，訓練 42B 個 token 的文本至少需要 58.3 小時。然而，在相同環境下基於本論文提出的方法，若使用 $\text{PPDB}_{\text{syn+ant}}$ 調整詞向量，每個 epoch 僅需 28 秒；若使用 $\text{WordNet}_{\text{syn+ant}}$ 調整詞向量，每個 epoch 僅需 7 秒；若使用 $\text{FrameNet}_{\text{syn}}$ 調整詞向量，每個 epoch 約需不到 1 秒，因此我們認為相較

[1] http://mattmahoney.net/dc/textdata.html

[2] https://ark.intel.com/content/www/tw/zh/ark/products/92993/intel-xeon-processor-e5-2603-v4-15m-cache-1-70-ghz.html

於使用大量文本訓練詞向量，使用本論文提出的方法調整使用適量文本訓練的詞向量更為有效率。

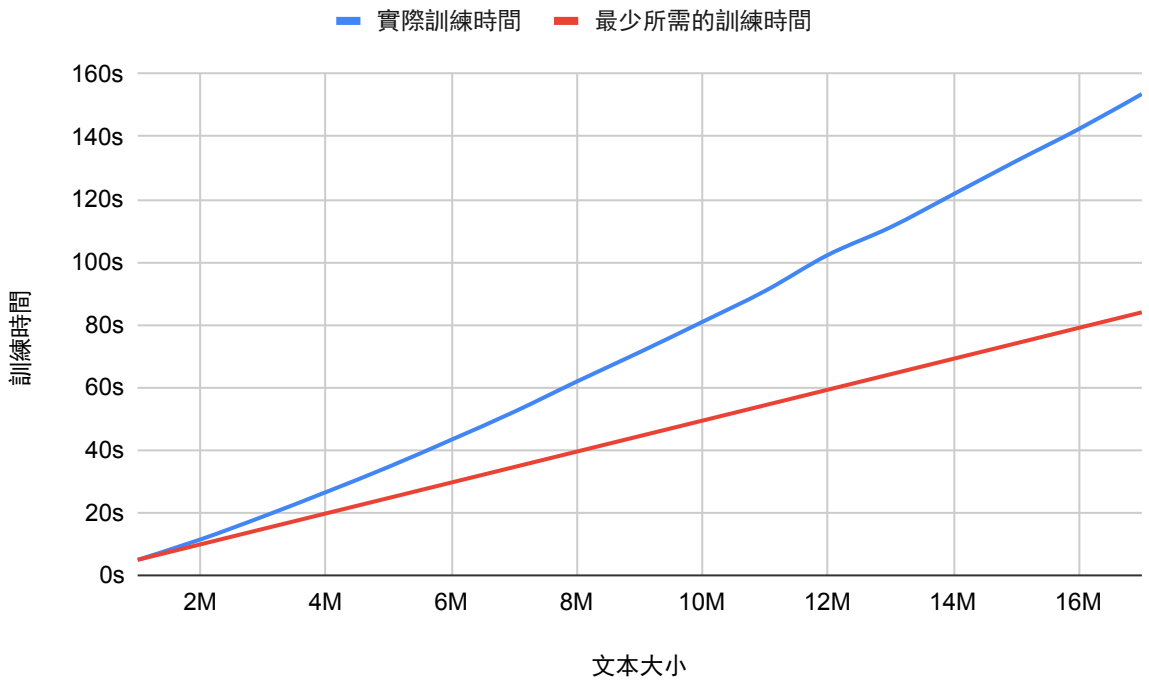
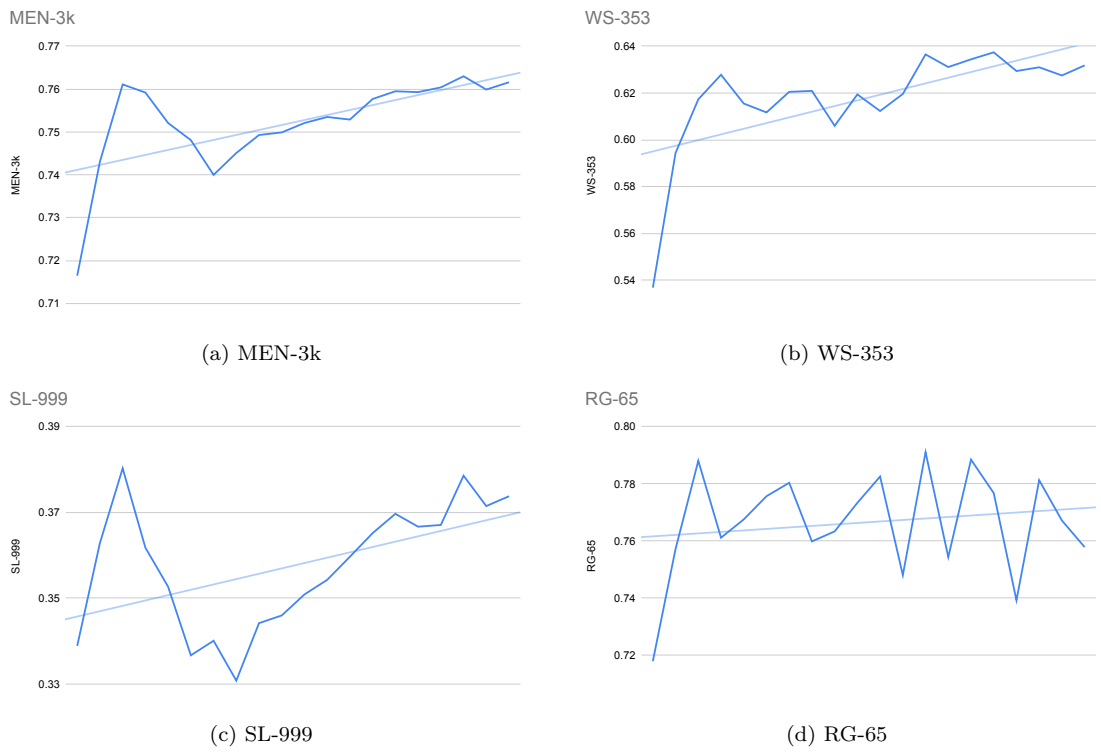


圖 5.1: 詞向量訓練時間

5.6 模型收斂狀況

為了檢驗模型是否有效的學習詞義資訊，本實驗記錄了模型在 0～20 個 epoch 的測試分數變化，結果如圖 5.2。從圖中的趨勢線我們可以發現所有任務都是有慢慢在進步的，且我們推測模型大多於第 2 個 epoch 找到局部最佳解，再朝向全局最佳解逼近。



(a) MEN-3k

(b) WS-353

(c) SL-999

(d) RG-65

圖 5.2: 模型收斂狀況

5.7 與相關研究比較

本實驗與相關研究比較，結果如表 5.12，其中由於 [12] 沒有開源，無法取得該論文中所使用的語義程度資料集，故難以複製實驗，故無法比較。而 [11][3] 雖有開源，但由於環境問題，程式經過修改仍無法執行，故也無法複製實驗。因此，我們比較的對象僅有代表 pairwise 方法的 [10][4] 和使用線性判別分析降維的 [13][5]，以下實驗數據均使用該論文開源的程式碼調整本論文使用的開源資料集，遺憾的是我們無法複製出與各論文中一樣的結果，推測的原因包含本論文用以過濾常見字的資料集 [6] 雖同 [11] 和 [13] 但不同於 [10]，且本實驗參數設定均使用程式預設值，可能並非最佳。

[3]https://github.com/nmrksic/counter-fitting
[4]https://github.com/mfaruqui/retrofitting
[5]https://github.com/HwiyeolJo/Extrofitting
[6]https://github.com/hermitdave/FrequencyWords/

表 5.12: 與相關研究比較

	MEN-3k	SL-999	WS-353	RG-65
GloVe_{6B}	0.7486	0.3692	0.6086	0.7693
Our Work	0.7612 ± 0.0031	0.3751 ± 0.0020	0.6334 ± 0.0084	0.7769 ± 0.0190
Retrofitting[10]	0.5089	0.3076	0.3112	0.4990
Extrofitting[13]	0.6889 ± 0.0007	0.3596 ± 0.0017	0.5780 ± 0.0014	0.7242 ± 0.0034

5.8 調整結果展示

本章節展示詞向量在經本論文與相關研究調整前後在相似詞上的範例與其視覺化，結果如表 5.13、表 5.14、圖 5.3、圖 5.4、圖 5.5、圖 5.6。

於表 5.13中，我們展示了 “easy” 在調整前後的相似詞，可以看到在調整後 “difficult” 已不在其相似詞中，但我們仍保留了其與其他相似詞之間的關係，這是理想的結果。但於表 5.14中，我們展示了 “short” 在調整前後的相似詞，可以看到在調整前後並沒有太大的差異，這是並不是理想的結果。

於圖 5.3、圖 5.4、圖 5.5、圖 5.6中，我們展示了 “man” 在調整前後相似詞的視覺化，可以看到在使用我們的方法調整後，我們認為除了相似詞較調整前符合人類的理解外，我們的方法非常好的保留了詞與詞之間的關係，例如 “he”、“his”、“him” 等詞較為接近，“father”、“uncle”、“brother”、“son”、“husband” 等詞較為接近。

表 5.13: 調整結果展示 (一)

	Top-6 similar words of “easy”					
GloVe	easier (.7287)	quick (.6607)	way (.6256)	make (.6243)	difficult (.6236)	simple (.6062)
Retrofitting	simple (.8034)	proved (.7491)	forceful (.7346)	making (.7051)	seemed (.7014)	easily (.7004)
Extrofitting	easier (.5226)	quick (.5165)	accessible (.3918)	easily (.3862)	straightforward (.3833)	simple (.3832)
Our Work	easier (.6870)	quick (.5939)	convenient (.5437)	simple (.5405)	make (.5185)	way (.5062)

表 5.14: 調整結果展示 (二)

	Top-6 similar words of “short”					
GloVe	long (.6962)	shorter (.5822)	length (.5626)	longer (.5618)	few (.5042)	only (.4987)
Retrofitting	birds (.6680)	pregnancy (.6590)	irrationally (.6460)	secretions (.6380)	eligible (.6368)	ordinarily (.6363)
Extrofitting	long (.5551)	shorter (.4423)	length (.3805)	duration (.3391)	ends (.3259)	mercifully (.3256)
Our Work	long (.6526)	shorter (.5761)	length (.4913)	longer (.4891)	end (.4178)	though (.4178)

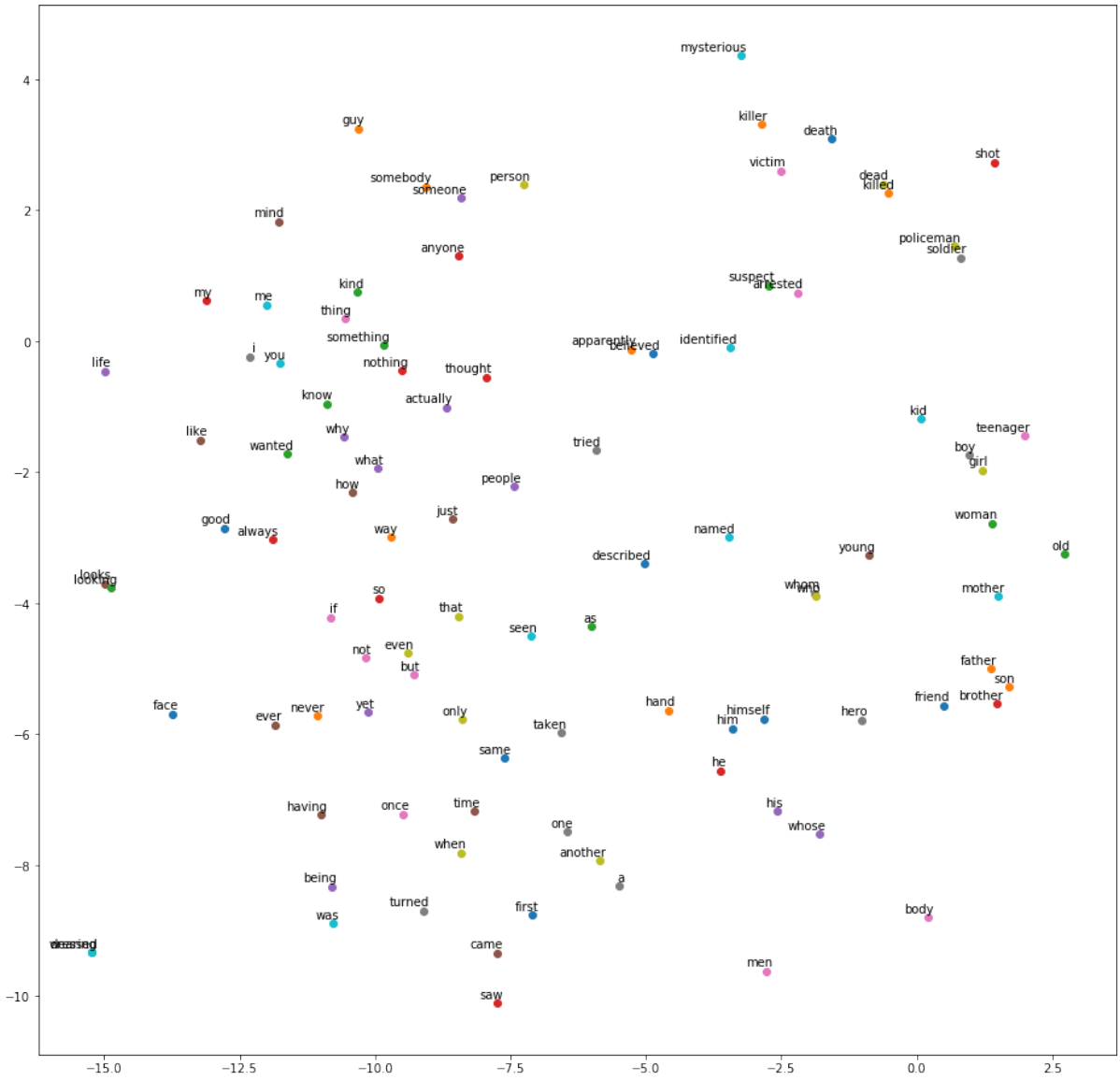


圖 5.3: Top-100 similar words of “man” in GloVe

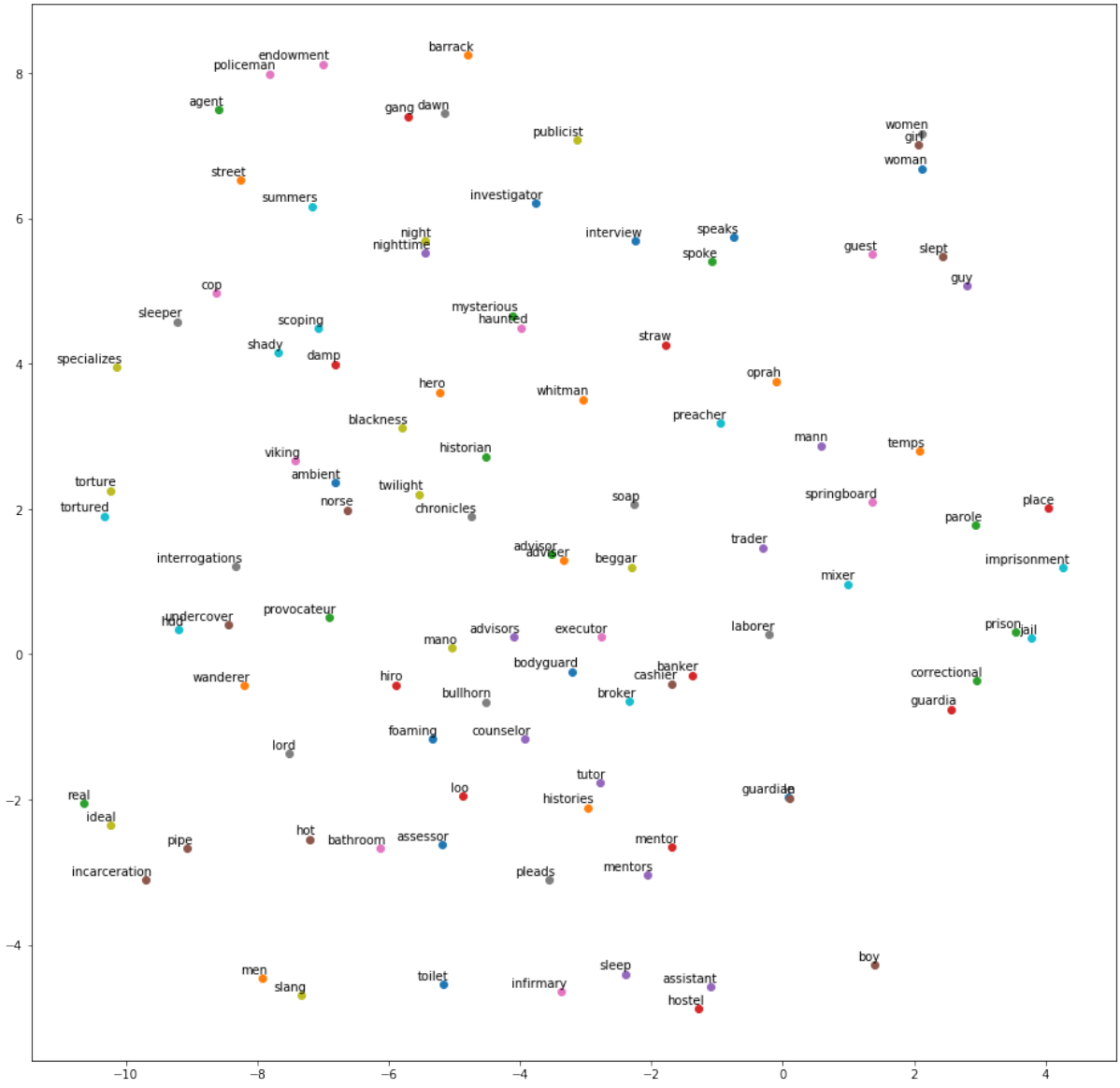


圖 5.4: Top-100 similar words of “man” in Retrofitting

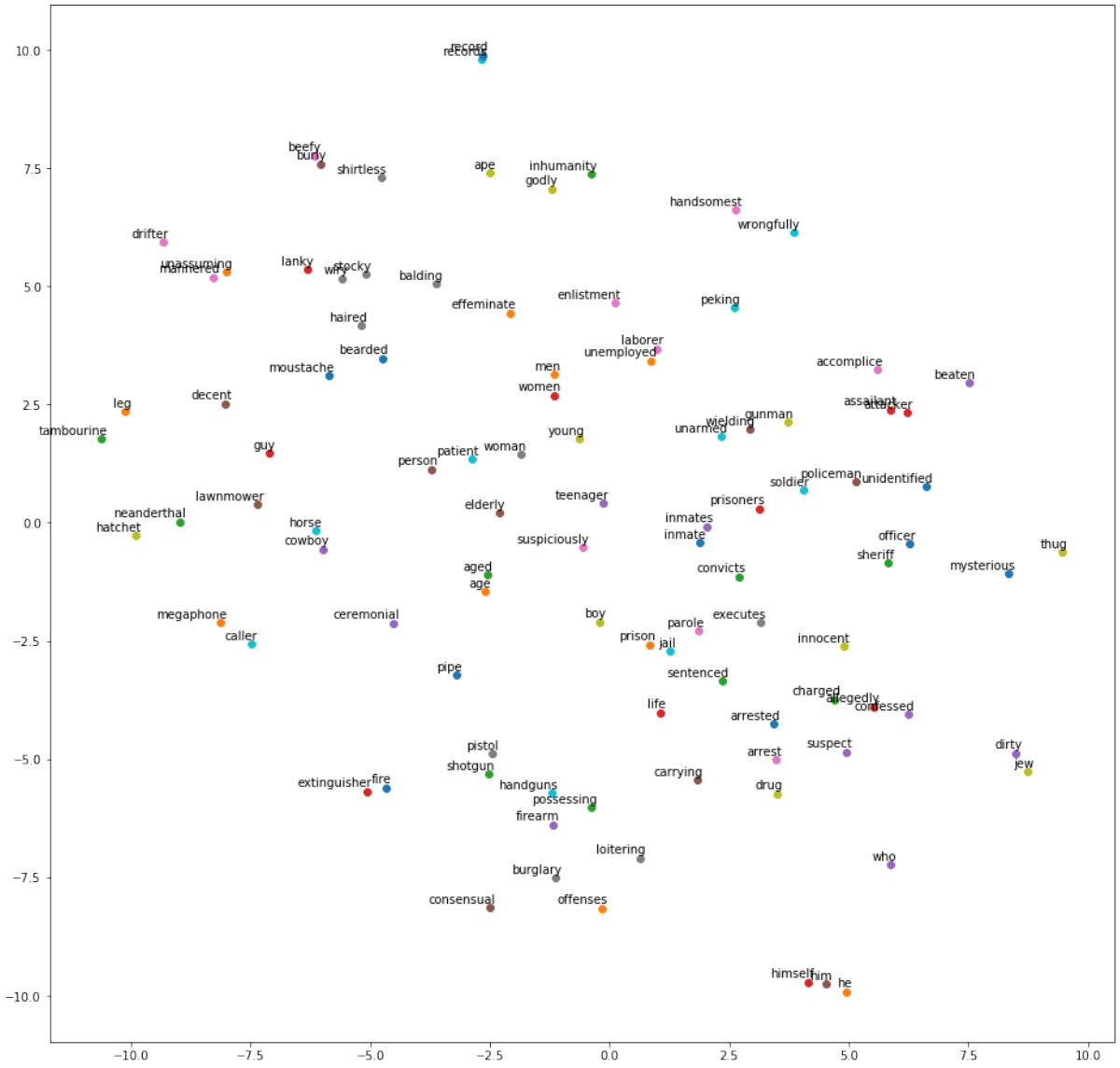


圖 5.5: Top-100 similar words of “man” in Extrofitting

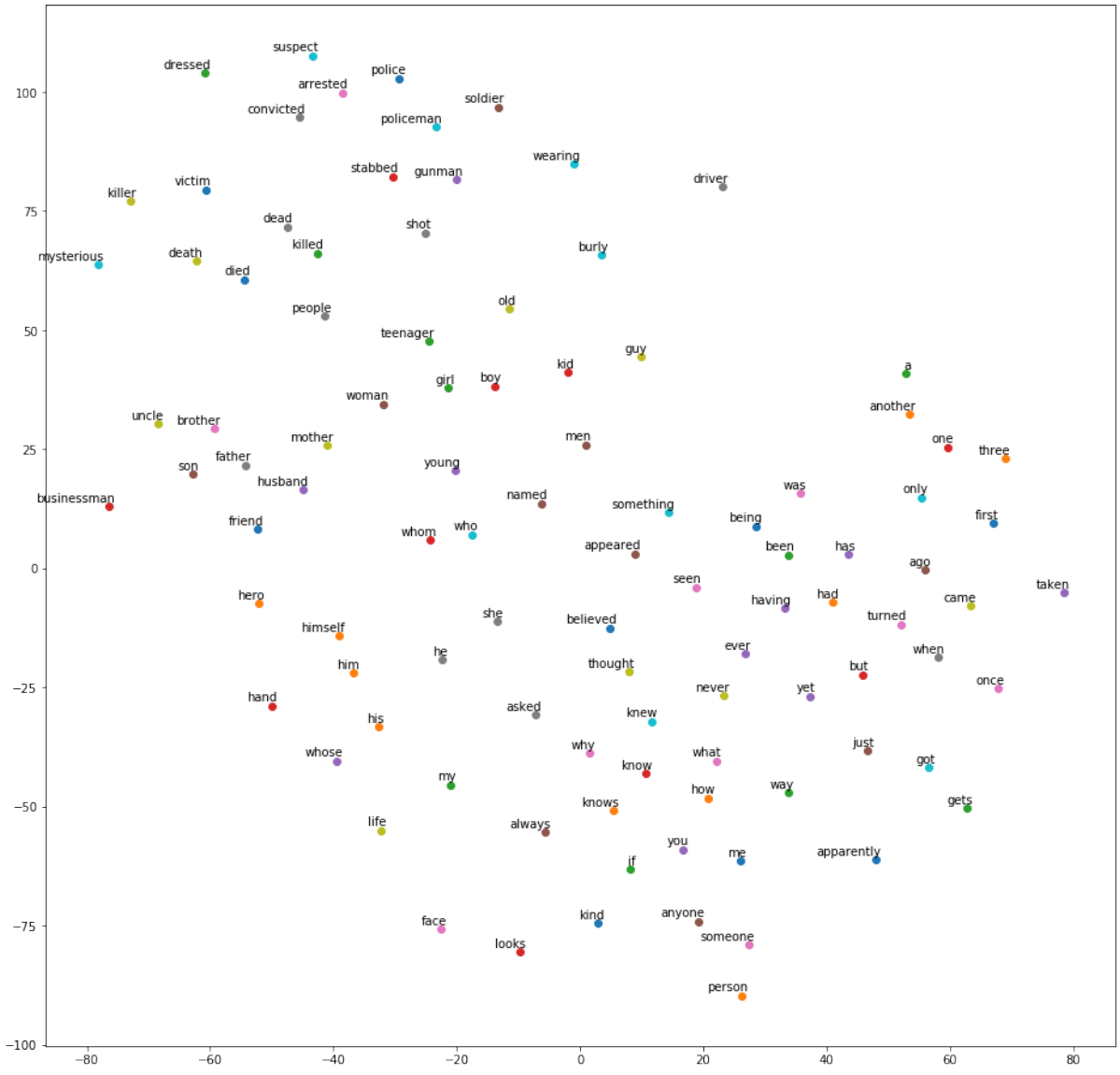


圖 5.6: Top-100 similar words of “man” in our work

六、 總結

6.1 結論

本論文提出了 listwise 的方法，使用自注意力機制做為編碼器，在訓練後使用從字典中萃取出的同義詞與反義詞資訊調整詞向量，提升詞向量的品質。

為驗證本方法的效果，我們設計實驗證明使用本方法加入從字典中萃取出的知識調整後，使用少量文本預訓練的詞向量在同義詞任務中表現可以超越使用大量文本預訓練的詞向量。除此之外，透過實驗結果我們還可以發現同義詞相較於反義詞在相似度任務上是更有用的資訊，且同義詞和反義詞資訊並不是越多越好，品質也會影響調整後的結果。

為驗證本方法使用的模型是否有效，我們設計實驗比較其與採用了全局資訊的線性轉換模型和只採用局部資訊的一維卷積模型，實驗結果顯示採用全局資訊的方法均優於只採用局部資訊的方法，且分配不同注意力在上下文中不同的詞上，再結合這些資訊調整詞向量更能有效的利用全局資訊。除此之外，為了更多的提升調整後詞向量的品質，本論文實驗了多頭式自注意力機制、深層的神經網路、正規化、殘差連結等方法，實驗結果後兩者效果顯著，但前兩者效果並不顯著。

6.2 未來展望

我們認為雖然本論文提出的方法應用於同義詞任務上有效，但仍需應用於更多的下游任務，來驗證將人類編撰的知識加入詞向量後，對詞向量是否真的具正面影響。

此外，我們認為本論文提出的方法效果仍有提升的空間，本論文透過實驗發現相對於同義詞資訊的數量，同義詞資訊的品質會更多的影響調整的結果，因此如何萃取出高品質的同義詞與反義詞資訊，我們認為是提升本方法效果的關鍵之一。而在目標函式方面，我們認為除了在保留原向量空間資訊的部分仍有進步空間外，加入如 [12] 使用的語義程度資訊等額外知識也機會提升本方法的效果。

最後，我們認為本論文提出的方法可以延伸至所有用到嵌入 (embedding) 或是向量 (vector) 來表示實體的應用，例如在電商平台可以使用商品類型資訊來調整表示商品的向量，或是用使用者資訊來調整表示使用者的向量等，因此希望未來能實作不同領域的應用。

參考文獻

[1] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[2] M. Baroni, G. Dinu, and G. Kruszewski, "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)*, Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 238–247. DOI: 10.3115/v1/P14-1023. [Online]. Available: https://www.aclweb.org/anthology/P14-1023.

[3] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. [Online]. Available: https://www.aclweb.org/anthology/D14-1162.

[4] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2013, pp. 3111–3119. [Online]. Available: http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

[5] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[6] M. Yu and M. Dredze, "Improving lexical embeddings with semantic knowledge," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 545–550. DOI: 10.3115/v1/P14-2089. [Online]. Available: https://www.aclweb.org/anthology/P14-2089.

[7] C. Xu, Y. Bai, J. Bian, B. Gao, G. Wang, X. Liu, and T.-Y. Liu, "Rc-net: A general framework for incorporating knowledge into word representations," in *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, 2014, pp. 1219–1228.

[8] J. Bian, B. Gao, and T.-Y. Liu, "Knowledge-powered deep learning for word embedding," in *Joint European conference on machine learning and knowledge discovery in databases*, Springer, 2014, pp. 132–148.

[9] D. Fried and K. Duh, "Incorporating both distributional and relational semantics in word representations," *arXiv preprint arXiv:1412.4369*, 2014.

[10] M. Faruqui, J. Dodge, S. K. Jauhar, C. Dyer, E. Hovy, and N. A. Smith, "Retrofitting word vectors to semantic lexicons," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Denver, Colorado: Association for Computational Linguistics, May 2015, pp. 1606–1615. DOI: 10.3115/v1/N15-1184. [Online]. Available: https://www.aclweb.org/anthology/N15-1184.

[11] N. Mrkšić, D. Ó Séaghdha, B. Thomson, M. Gašić, L. M. Rojas-Barahona, P.-H. Su, D. Vandyke, T.-H. Wen, and S. Young, "Counter-fitting word vectors to linguistic constraints," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 142–148. DOI: 10.18653/v1/N16-1018. [Online]. Available: https://www.aclweb.org/anthology/N16-1018.

[12] J.-K. Kim, M.-C. de Marneffe, and E. Fosler-Lussier, "Adjusting word embeddings with semantic intensity orders," in *Proceedings of the 1st Workshop on Representation Learning for NLP*, Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 62–69. DOI: 10.18653/v1/W16-1607. [Online]. Available: https://www.aclweb.org/anthology/W16-1607.

[13] H. Jo and S. J. Choi, "Extrofitting: Enriching word representation and its vector space with semantic lexicons," in *Proceedings of The Third Workshop on Representation Learning for NLP*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 24–29. DOI: 10.18653/v1/W18-3003. [Online]. Available: https://www.aclweb.org/anthology/W18-3003.

[14] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., 2017, pp. 5998–6008. [Online]. Available: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

[18] I. Yamada, A. Asai, J. Sakuma, H. Shindo, H. Takeda, Y. Takefuji, and Y. Matsumoto, "Wikipedia2vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from wikipedia," *arXiv preprint 1812.06280v3*, 2020.

[19] E. Pavlick, P. Rastogi, J. Ganitkevitch, B. Van Durme, and C. Callison-Burch, "PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 425–430. DOI: 10.3115/v1/P15-2070. [Online]. Available: https://www.aclweb.org/anthology/P15-2070.

[20] S. Rajana, C. Callison-Burch, M. Apidianaki, and V. Shwartz, "Learning antonyms with paraphrases and a morphology-aware neural network," in *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 12–21. DOI: 10.18653/v1/S17-1002. [Online]. Available: https://www.aclweb.org/anthology/S17-1002.

[21] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995, ISSN: 0001-0782. DOI: 10.1145/219717.219748. [Online]. Available: https://doi.org/10.1145/219717.219748.

[22] C. F. Baker, C. J. Fillmore, and J. B. Lowe, "The Berkeley FrameNet project," in *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, Montreal, Quebec, Canada: Association for Computational Linguistics, Aug. 1998, pp. 86–90. DOI: 10.3115/980845.980860. [Online]. Available: https://www.aclweb.org/anthology/P98-1013.

[23] E. Bruni, N. K. Tran, and M. Baroni, "Multimodal distributional semantics," *J. Artif. Int. Res.*, vol. 49, no. 1, pp. 1–47, Jan. 2014, ISSN: 1076-9757.

[24] F. Hill, R. Reichart, and A. Korhonen, "SimLex-999: Evaluating semantic models with (genuine) similarity estimation," *Computational Linguistics*, vol. 41, no. 4, pp. 665–695, Dec. 2015. DOI: 10.1162/COLI_a_00237. [Online]. Available: https://www.aclweb.org/anthology/J15-4004.

[25] "Placing search in context: The concept revisited," *ACM Trans. Inf. Syst.*, vol. 20, no. 1, pp. 116–131, Jan. 2002, ISSN: 1046-8188. DOI: 10.1145/503104.503110. [Online]. Available: https://doi.org/10.1145/503104.503110.

[26] H. Rubenstein and J. B. Goodenough, "Contextual correlates of synonymy," *Commun. ACM*, vol. 8, no. 10, pp. 627–633, Oct. 1965, ISSN: 0001-0782. DOI: 10.1145/365628.365657. [Online]. Available: https://doi.org/10.1145/365628.365657.