



國立臺灣大學電機資訊學院資訊工程學研究所

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

以簡化法處理數學文字題

A Reduction Based Approach for  
Math Word Problem

黃孟霖

Meng-Lin Huang

指導教授：項潔博士、許聞廉博士

Advisor: Jieh Hsiang, Ph. D. and

Wen-Lian Hsu, Ph. D.

中華民國 109 年 08 月

August 2020





## 致謝

感謝項潔教授與許聞廉老師兩年來的諄諄教誨與照顧。感謝IASL Lab的學長姐給了一些論文的方向與幫助，感謝德堯、俊璿、偉倫與育婷予我對於論文上的啟發與人生上的談話，感謝女友在我充滿挑戰的研究路上的陪伴，感謝我的父母對於我讀研究所的這個決定的支持。轉眼間，碩士班寒窗苦讀的兩年生活恍若隔世，驀然回首，回憶如洪水潰堤般傾瀉而下，席捲而來的是無限的感謝，離去卻也讓我依依不捨，感謝所有的你們在這樣的歲月，與這樣的我度過這樣平凡幸福且瑣碎的日子。





## 摘要

數學文字題包含自然語言理解與邏輯推斷，所以一直以來都被視為人工智慧領域重要的應用，其中小學數學問題並沒有複雜的數學計算與文字敘述，研究人員能讓其設計出來的系統專注於語言理解與邏輯推斷的部分，而不需要涵蓋廣大的文法以及數學相關的Domain Knowledge，也因此非常適合用來當作衡量的數據集。

本篇論文提出一個新的小學數學解題系統，與現今其他以神經網路建構的系統的不同之處在於此系統著重於邏輯推斷過程。我們的系統除了可以算出答案以外還能產生出此答案的解釋，並且透過簡化題目來減少所需要設計的規則。此外針對此系統目前處理不了的問題我們也做了錯誤分析，並設計了新的句子簡化方法，更進一步的將其與系統進行結合，最終此系統在我們提出的小學數學數據集上得到顯著的成果。

關鍵字:數學文字問題，自然語言理解，句子過濾，邏輯推斷，深度學習





# Abstract

Math Word Problem involves natural language understanding and logical inference, thus it has been regarded as a major application of AI. Because there is no complicated mathematical calculation in elementary math word problem, researchers could concentrate on the task of understanding and inference.

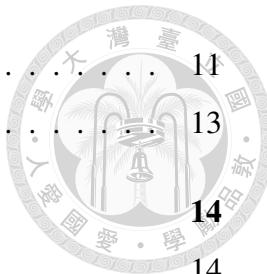
In this thesis, we propose a novel elementary math solver system, named “elementary mathematics problem solver”. The difference between prevailing neural network based systems and ours is that we focus more on the process of logical inference and domain common sense. Our solver not only answers the question, but also gives corresponding explanations. We adopt label sequence as “pattern” to identify similar sentences and problems. Solution strategy is described in natural language script so that teachers can make change at will. We greatly reduce the patterns needed by filtering out non-essential words in problem sentences. Since the explanations are written in natural language, error analysis is very intuitive. Our problem solving philosophy is based on mimicking how humans learn to solve a problem. Ultimately, our system achieves significant performance on our elementary mathematics dataset.

**Keyword :** math word problem , natural language understanding, sentence input filtering, logic inference, deep learning



# Contents

|   |    |
|---|----|
| 致謝  | ii |
| 摘要  | iv |
| <b>Abstract</b>   | vi |
| <b>List of Figures</b>                                      | ix |
| <b>List of Tables</b>                                       | x  |
| <b>1 緒論</b>   | 1  |
| 1.1 研究目的與動機 . . . . .                                       | 1  |
| 1.2 數學文字題 . . . . .   | 1  |
| 1.3 主要貢獻 . . . . .  | 2  |
| 1.4 章節概要 . . . . .  | 2  |
| <b>2 相關研究</b>   | 4  |
| 2.1 背景 . . . . .  | 4  |
| 2.1.1 循環神經網路(Recurrent Neural Network, RNN) . . . . .       | 4  |
| 2.1.2 長短期記憶模型(Long Short Term Memory Model, LSTM) . . . . . | 4  |
| 2.1.3 Sequence to Sequence模型 . . . . .                      | 5  |
| 2.1.4 Transformer . . . . .                                 | 6  |
| 2.1.5 TF-IDF . . . . .                                      | 8  |
| 2.1.6 Word2Vec . . . . .                                    | 9  |
| 2.1.7 BERT . . . . .  | 9  |
| 2.1.8 Ontology(本體論) . . . . .                               | 10 |
| 2.2 相關研究 . . . . .  | 11 |



|  |           |
|--|-----------|
| 2.2.1 歷史脈絡 . . . . .                       | 14        |
| 2.2.2 小學數學題解題系統 . . . . .                  | 13        |
| <b>3 資料集</b>                               | <b>14</b> |
| 3.1 問題描述與資料集 . . . . .                     | 14        |
| <b>4 小學數學解題系統</b>                          | <b>16</b> |
| 4.1 小學數學解題流程 . . . . .                     | 16        |
| 4.1.1 切句與簡化(reduction) . . . . .           | 17        |
| 4.1.2 對到框架並產生instancemap . . . . .         | 19        |
| 4.1.3 改寫(rewrite) . . . . .                | 20        |
| 4.1.4 跑script與整合解釋 . . . . .               | 21        |
| <b>5 句子過濾</b>                              | <b>24</b> |
| 5.1 通用過濾法 . . . . .                        | 24        |
| 5.1.1 Rouge-N based . . . . .              | 25        |
| 5.1.2 Transformer based . . . . .          | 25        |
| 5.1.3 Pre-trained BERT based . . . . .     | 28        |
| 5.1.4 Label Sequence . . . . .             | 31        |
| <b>6 實驗</b>                                | <b>34</b> |
| 6.1 小學數學解題系統答題統計及分析 . . . . .              | 34        |
| 6.2 各簡化機制實驗 . . . . .                      | 35        |
| 6.2.1 額外答對題數比較 . . . . .                   | 35        |
| 6.2.2 句子通順度比較 . . . . .                    | 38        |
| 6.2.3 通用簡化法 . . . . .                      | 39        |
| 6.2.4 Label Sequence加上小學數學解題系統分析 . . . . . | 40        |
| <b>7 結論與未來展望</b>                           | <b>42</b> |
| 7.1 結論 . . . . .                           | 42        |
| 7.2 未來展望 . . . . .                         | 42        |
| <b>Bibliography</b>                        | <b>44</b> |



# List of Figures

|     |   |    |
|-----|---|----|
| 2.1 | Sequence to Sequence 模型架構圖[1] . . . . .   | 5  |
| 2.2 | Tansformer模型架構圖[2] . . . . .  | 7  |
| 2.3 | Bert架構圖[3] . . . . .  | 10 |
| 2.4 | 簡易ontology範例 . . . . .  | 10 |
| 4.1 | 小學數學系統解題流程圖 . . . . .   | 17 |
| 4.2 | Reduction機制圖 . . . . .  | 18 |
| 4.3 | 小學數學系統中的框架架構圖 . . . . .   | 19 |
| 4.4 | 句子產生instance map的流程圖 . . . . .  | 20 |
| 4.5 | ScriptNL例子 . . . . .  | 22 |
| 4.6 | RNL轉化示意圖 . . . . .  | 22 |
| 5.1 | 詞性增減示意圖 . . . . .   | 26 |
| 5.2 | 刪減字數分佈圖，縱軸是佔的比率，橫軸是刪減的字數 . . . . .  | 27 |
| 5.3 | 模型架構示意圖 . . . . .   | 27 |
| 5.4 | Label Sequence 流程圖 . . . . .  | 32 |
| 5.5 | Label Sequence 加上小學數學解題系統流程圖 . . . . .  | 33 |
| 6.1 | 答對與答錯在題型上的分佈 . . . . .  | 35 |
| 6.2 | 各個方法刪除字數的分佈，左上是Rouge-N，右上是BERT based，<br>左下是Label Sequence，右下是Transformer based . . . . . | 37 |
| 6.3 | Transformer based在機率閾值設為0.6時的情況 . . . . .   | 38 |
| 6.4 | 左邊是舊的”分配”框架下的instance map，右邊是新設計的instance map   | 41 |



# List of Tables

|     |   |    |
|-----|---|----|
| 3.1 | 各出版社題型分佈 . . . . .  | 14 |
| 3.2 | 各題型例子 . . . . .   | 15 |
| 6.1 | 各個簡化方法的比較,I Rouge-N代表迭代Rouge-N算法 . . . . .  | 36 |
| 6.2 | 各簡化法刪減字比較，LS代表Label Sequence，R是Rouge-N，T是Transformer based，B是BERT based，紅字代表被刪除的字 . . . . . | 36 |
| 6.3 | 各個簡化方法通順度分數表 . . . . .  | 39 |



# Chapter 1

## 緒論

### 1.1 研究目的與動機

在人工智能發達的現在，我們期望有一套通用的人工智能系統可以解決所有問題，然而這套系統現在離我們的生活還很遙遠，於是人們轉而去尋求一個能確切達成某個小目標的系統，而如何去衡量人工智能的智能的部分，考試或許是一個最直觀的解答。早在約莫1500年前古代的數學著作《孫子算經》中記載了下列問題，“今有雉兔同籠，上有三十五頭，下有九十四足，問雉兔各幾何？”這就是家喻戶曉的雞兔同籠問題，也是現今大部分的小學生都能解決的問題，如果人工智能能透過閱讀文字敘述的題目，進而推敲出算式，或許能理解為機器學習到了某樣知識，並了解題目意思，也就是智能的具現化。因此數學文字題(Math Word Problem)是用來測量機器理解與否最適當的題目類型。

### 1.2 數學文字題

數學文字題(Math Word Problem)為給定一段文字敘述，列出式子算出答案，其中牽涉語言理解與邏輯推斷，解題系統甚至要能抓出題目細微的差異，舉例而言以下的例子：

水流時速 $5\text{km/hr}$ ，船速 $20\text{km/hr}$ ，船順流而行，兩小時可航行多少公里?  
(1.1)

水流時速 $5\text{km/hr}$ ，船速 $20\text{km/hr}$ ，船逆流而行，兩小時可航行多少公里?

系統要能抓到逆流跟順流與船速跟水速之間的關係，也因此必須完全了解題目的意思，因為包含高度語言理解，所以數學文字問題一直以來都是自然語言處理(Natural Language Process)裡熱門的問題，現今主流的作法是透過sequence to



sequence model針對每個問題輸出其對應的算式，然後再算出答案，但這種類神經網路(Neural Network)為基礎的系統有一個原生的問題，從數學文字題目的理解到產生算式的過程全然是黑盒子(black-box)，我們難以理解模型其中的推論過程，因此很難生出解釋，也很難對其錯誤進行改進。

於是有所別於以往的做法，在這篇論文中我們提出一個以過濾與簡化法簡化小學數學題，並透過對問題句子進行語意框架的分析，再個別對分析的結果產生出算式與解釋，最終輸出結果的小學數學解題系統。透過此方法建構出的系統比起Neural Network based的系統更具可解釋性與可修改性。

此外對於系統尚未能處理的問題我們做了幾點分析，歸納出兩種錯誤方式，即為語意框架尚未建構或者是語意框架已建構但未對應到，對於後者本篇論文也提出了三種改進方式，利用前處理的方式提升系統解題的可變通性。

### 1.3 主要貢獻

- 提出一個以Reduction為主的小學數學解題系統，能進行數學計算也能做邏輯推斷
- 針對系統目前尚不能解的題目，提出數個前處理的方式來改善
- 設計一個可行的方案，融合改善方法與舊有的系統

### 1.4 章節概要

本篇論文章節整理如下

- Chapter 2 - 背景

本章節會簡略的複習關於本篇論文所需要的先備知識以及統整近幾年來針對數學文字問的研究，與所遇到的挑戰

- Chapter 3 - 資料集

本章節介紹本篇論文中所使用到的資料集與問題描述



- Chapter 5 - 小學數學解題系統  
本章節鉅細彌遺的先容我們所提出的解題系統
- Chapter 6 - 相似句比對  
本章節包含了數個提出比較相似框架的方法
- Chapter 7 - 實驗與錯誤分析  
本章節輔以研究數據來對我們的系統與方法進行評估
- Chapter 8 - 總結與未來展望  
本章節總結目前的研究，並針對未來可以改進的方面提出構思



# Chapter 2

## 相關研究

本章節會闡述論文與相關研究中運用的知識與技術。

### 2.1 背景

#### 2.1.1 循環神經網路(Recurrent Neural Network, RNN)

循環神經網路[4]，被用來處理有時間資訊的序列資料，RNN會根據序列資料 $(x_1, x_2, \dots, x_t)$ 產生其對應的隱藏層(Hidden Representation) $(h_1, h_2, \dots, h_t)$ ，隱藏層是基於以下公式遞迴而來：

$$h_t = f(W_x x_t + W_h h_{t-1} + b_h) \quad (2.1)$$

$f$ 可以是任何一種非線性激活函數，不然多個線性函數的組合也還只是一個大的線性函數， $W_x$ 和 $W_h$ 是模型的權重(Weight)， $b_h$ 是模型的偏值(bias)，兩者皆是透過學習來進行更新。在每個時間點， $h_t$ 通常會額外通過個線性投影層(Linear Projection layer)輸出預測值 $o_t$ 。舉例而言如果我們希望去預測的是下個時間點會出現的字，那 $o_t$ 就會是在詞典中每個字出現的機率分佈，公式如下：

$$o_t = softmax(W_o h_t + b_o) \quad (2.2)$$

#### 2.1.2 長短期記憶模型(Long Short Term Memory Model, LSTM)

當時間序列的資料過於長時，上一節所提到的RNN就會面臨梯度爆炸(gradient explosion)以及梯度消失(gradient vanishing)的問題，造成前者的主因是在利用鏈式法則(Chain Rule)進行反向傳播(back propagation)進行模型參數更新



時，當模型乘上大於1的值數次以後，且時間序列很長時，前面時間點會接受到非常大的gradient，以致每次更新影響結果很大，導致訓練不穩定；而後者是當模型乘上小於1的值數次前面時間點會接受到非常小的gradient，以致於難以對其進行更新。於是長短期記憶元件[5](Long Short Term Memory Cell)被提出來處理此問題，LSTM每個時間點的更新變為如下：

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2.3)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2.4)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (2.5)$$

$$C_t = f * C_{t-1} + i_t * \tilde{C}_t \quad (2.6)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (2.7)$$

$$h_t = o_t * \tanh(C_t) \quad (2.8)$$

LSTM的特色在於引入了三個gates，遺忘閥(forget gate)用來表示要遺忘前些時間點的記憶的比率，輸入閥(input gate)代表現在這個時間點的資訊有多少要被記住，輸出閥(output gate)為考慮到當前時間點包含以前的資訊後對於輸出的貢獻度有多少，三者的值都是透過前個時間點的隱藏層 $h_t$ 與當前輸入 $x_t$ 來決定， $W_f$ 、 $W_i$ 、 $W_o$ 、 $W_c$ 、 $U_f$ 、 $U_i$ 、 $U_o$ 、 $U_c$ 、 $b_f$ 、 $b_i$ 、 $b_o$ 、 $b_c$ 是模型要去學的權重與偏值， $\sigma$ 是非線性激活函數輸出介於0與1之間的機率值，通常為sigmoid函數。

### 2.1.3 Sequence to Sequence模型

Sequence to Sequence(seq2seq)[1]模型算是以上幾種時間序列模型最典型的應用，舉凡機器翻譯、摘要、對話系統都可以見到seq2seq模型的蹤跡。seq2seq模型架構如圖2.1

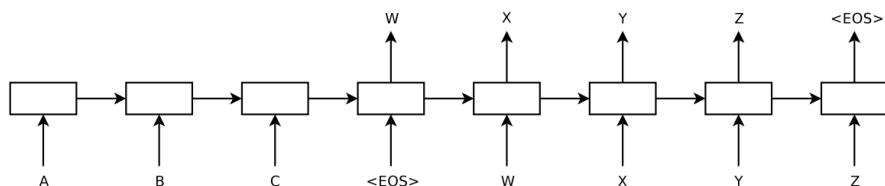


圖 2.1: Sequence to Sequence 模型架構圖[1]

seq2seq模型由一個編碼器(encoder)跟一個解碼器(decoder)組合而成，編碼器跟解碼器分別是時間序列模型，編碼器的輸入是時間序列的資料加上隨機初始的



隱藏向量 $h$ ，解碼器的輸入依照有沒有teacher forcing而定，可能是上個時間的模型輸出或是真實label，隱藏向量則為編碼器最後的輸出。舉例而言若是做機器翻譯問題，則編碼器跟解碼器分別為RNN模型，編碼器輸入為原句子(要被翻譯的句子)與隨機初始的隱藏向量 $h$ ，解碼器的輸入是目標句子(真實翻譯後的句子)與編碼器最後輸出的隱層向量，並且只計算解碼器的損失函數，透過反向傳導傳給編碼器。

seq2seq模型的原理實際上就是希望在做解碼時能記住編碼的結果，以中翻英為例，生出的英文翻譯是根據中文句子為基底，seq2seq模型透過隱藏層傳遞的結果來達到這樣的條件機率。

#### 2.1.4 Transformer

RNN-based的模型產生每個時間點的隱藏層 $h_t$ 時必須仰賴上個時間點的隱藏層 $h_{t-1}$ ，這樣的結果導致了RNN本質上無法進行平行運算，在長序列時所花費的時間不貲。與此相對而言，CNN-based[6]的模型天生適合平行運算但排斥序列型的計算。於是在2017時Google提出Transformer[2]模型，以注意力(Attention)機制為主軸進行平行運算，完全捨棄遞迴(recurrence)的架構。

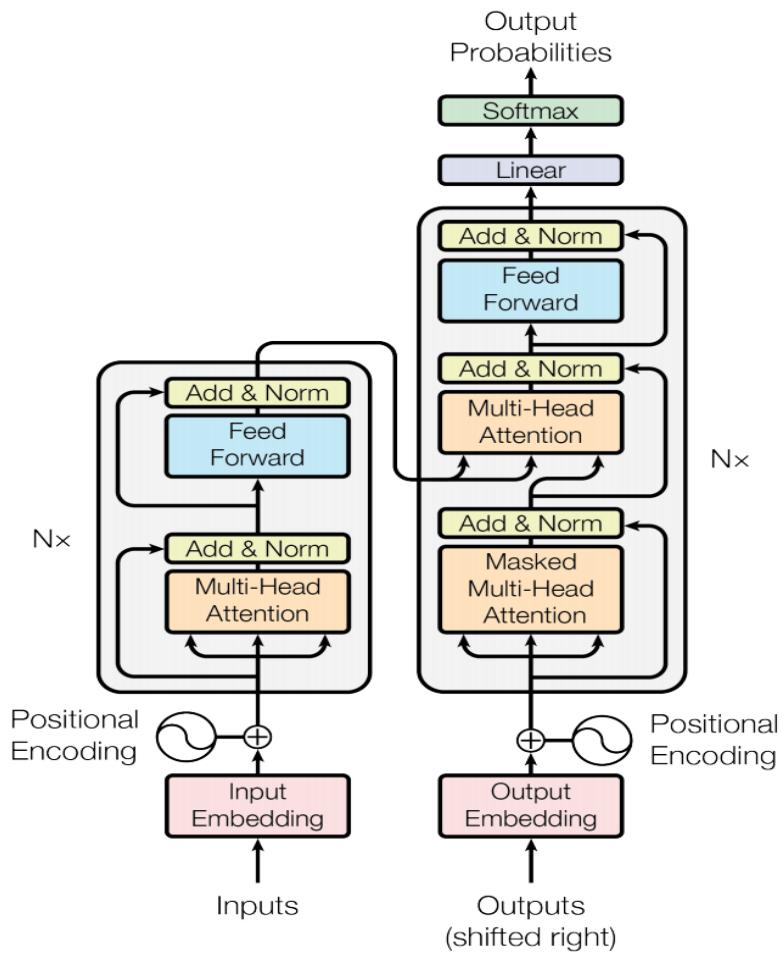


圖 2.2: Transformer模型架構圖[2]

Transformer架構如圖2.2，Head Attention即是每個字考慮到其他所有字後得到權重的過程，做法上是每個字都有自己的query, key, value。透過自己的key與每個其他字的value算出一個權重，這個權重代表的就是自己與每個字的相關程度，將所有權重乘上自己的value得到加權輸出。而Transformer包含Multi-Head，就是把多個這樣的Head Attention的輸出通過併接(Concatenate)的方式合成起來得到最終的輸出，思路是預期每個Head能注意到每個字之間不同面向的關聯性，公式如下：

$$\begin{aligned} \text{Attention}(Q, K, V) &= \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}V\right) \\ \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^o \quad (2.9) \\ \text{where } \text{head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned}$$

其中參數  $W_i^Q \in R^{d_{model} \times d_k}, W_i^K \in R^{d_{model} \times d_k}, W_i^V \in R^{d_{model} \times d_v}$  以及  $W^O \in R^{hd_v \times d_{model}}$  Transformer捨棄了遞迴(recurrence)的架構，但位置的資訊對於包含

時間序列的資料而言還是非常重要，所以Tranformer中引入了位置編碼(Positional Encoding)期望再一次的將時間資訊融入模型中，具體公式如下：



$$\begin{aligned} PE_{(pos,2i)} &= \sin(pos/10000^{(2i/d_{model})}) \\ PE_{(pos,2i+1)} &= \cos(pos/10000^{(2i/d_{model})}) \end{aligned} \quad (2.10)$$

$pos$ 是該字的位置， $i$ 是向量的維度，也就是每一個維度的值是一個正弦(sin)或是餘弦(cos)函數，這樣的設計是因為原作者認為距離k個位置遠的字的位置編碼( $PE_{pos+k}$ )應該可以視為當前位置的位置編碼( $PE_{pos}$ )的線性組合。位置編碼跟輸入編碼(input Embedding)有著相同的維度，因此加兩者相加得到富含位置資訊最終的輸入編碼。

此外會加入正規化層(Normalization Layer)，正規化的目的在於希望每一個批次(Batch)數據彼此之間的大小不會差太多，舉例而言若有個數據資料包含 $x_1 = 1$ 和 $x_2 = 100$ 這兩筆數據，那在更新時同樣的權重 $w$ 對於彼此的影響就會相差很多，這會讓我們的訓練過程變得太不容易，所以引入了批次正規化的技術，讓每次資料之間彼此的差距縮小，可以訓練的更好且更容易收斂，實務上在執行時我們只需要讓每筆資料減去平均值除以變異數即可，公式如下：

$$\tilde{z}_i = \frac{z_i - \mu}{\sigma} \quad (2.11)$$

### 2.1.5 TF-IDF

在NLP領域中我們希望把文字轉成程式可以處理的型式，其中一種想法是以詞在每篇文章中跟所有文章中的出現頻率當作這個詞對於資料集的特徵，此方法稱為TF-IDF。

TF(Term Frequency)概念上指的就是詞在文章中出現的次數，會是 $N*D$ 維的矩陣，其中 $N$ 是辭典大小， $D$ 是文章總數，矩陣中的每一項是第 $N$ 個字在 $D$ 份文章裡出現的次數除以對所有詞在 $D$ 份文章中的次數加總。IDF(inverse document frequency)逆向文件頻率，公式如2.12， $d_x$ 指的是詞 $x$ 在幾篇文章中出現過， $D$ 是文章總數。

$$idf_t = \log\left(\frac{D}{d_x}\right) \quad (2.12)$$

最後TF-IDF就是把兩項結合起來，也就是每個詞對每篇文章的分數，公式如2.13

$$score_{x,d} = tf_{x,d} * idf_x \quad (2.13)$$



### 2.1.6 Word2Vec

另外一種簡單且直觀的想法是對辭典大小建一個one-hot-encoding的編碼，然而這種方式的編碼會遇到一個問題就是詞與詞之間是沒有關聯性的，所以Tomas Mikolov等人提出了以N維的向量來表示一個字，也就是俗稱的Word2Vec[7]，而這個也是現今主流的方式，做法上有分成兩種，CBOW與Skip-gram。CBOW是給定上下文預測中間的字，Skip-gram是給定中間字預測上下文。

然而Word2Vec有個極大的缺陷，即它是靜態的向量，以英文”bank”為例，有河岸或是銀行的意思，但由於Word2Vec對於每個字都只會有一個編碼，也就是在為河岸的意思時卻還會帶有銀行的意思，這顯然是不太合邏輯的，所以我們希望的是根據上下文來產生相對應的詞向量，於是後來就有人提根據上下文有不同詞向量的作法，稱為Contextual Embedding，其中就在2018年Google AI research提出以Transformer為主軸的BERT [3]開出了之後百家爭鳴的序幕。

### 2.1.7 BERT

BERT架構如Figure2.3只是Transformer的延伸，BERT創新的點在於預訓練階段以及可以良好地結合下游任務。預訓練方面BERT將一個句子中15%的字詞屏蔽掉，在算損失時我們只需要去計算那些被屏蔽掉的字即可，第二個即是做下句預測，也就是去預測輸入的兩個句子是否為上下句，兩句之間以特殊符號”[SEP]”做分隔，這樣的預訓練方式在邏輯上可以想成當我們有個良好的語言模型它要能做到讀懂句子的意思，所以判斷上下句可以視為一個理解句子意思的依據，同理能填補被屏蔽的字也算是充分了解句子意思的標竿。

在結合下游任務方面，除了可以直接拿BERT輸出的contextual embedding當作詞向量輸入放進下游模型以外，還能做問答句子分類單詞分類，以句子分類為例，BERT論文中直接將句首的符號”[CLS]”的輸出向量過一個線性分類器即可，單詞分類則是將每個單詞的輸出過一個線性轉換，而問答是將問題分別與每個答案接起來，中間以”[SEP]”做區隔，最後算出分數做排序。

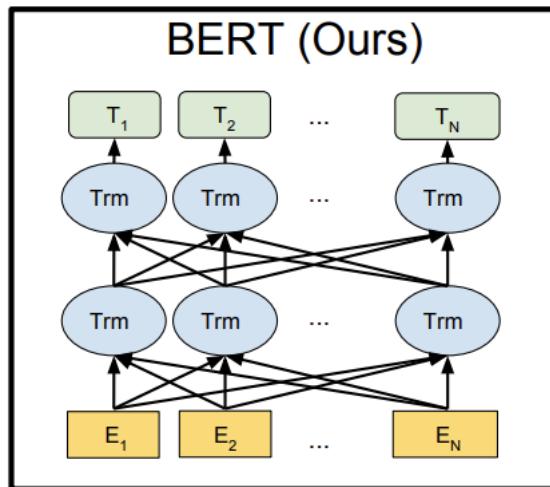


圖 2.3: Bert架構圖[3]

### 2.1.8 Ontology(本體論)

本體指的是對於特定領域之中某套概念及其相互之間關係的形式化表達，本體是人們以自己領域的知識依照特定方式編寫出來的原理或是事實，本體可用來對該領域的屬性進行推理，也可以用來定義該領域的屬性。

圖2.4是簡易的本體範例:

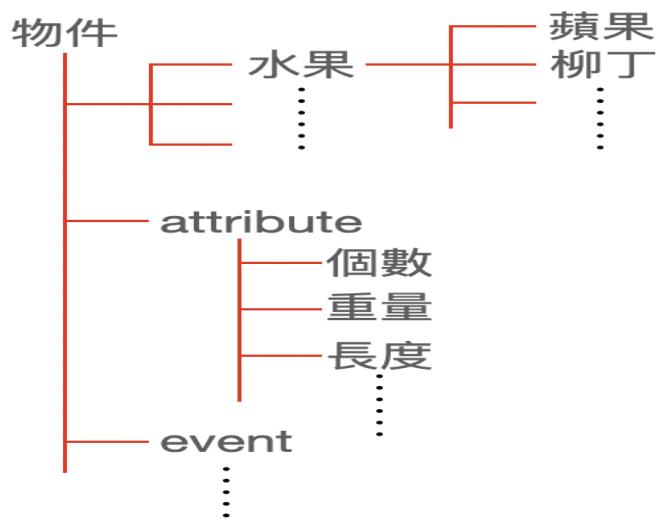


圖 2.4: 簡易ontology範例



以圖中的例子是我們透過對物件這詞相關知識建構出來的關係圖，對於一個物件我們會在意是什麼物件，這個物件的屬性例如數量長度，以及是什麼樣的事件作用在此物件上，在小學數學中可能是”被買”、“被吃”或是”被擁有”之類的關係，總而言之本體論即是各個領域的學者運用自己的專業知識，建構如Figure2.4的關係圖。

## 2.2 相關研究

### 2.2.1 歷史脈絡

自動數學文字題處理最早可追溯到1964年Bobrow[8]提出的STUDENT系統，做法上是輸入有限定描述方式的數學題目，並且人工定義幾組關係詞與關係，如”equal”，”sum”、”product”，把句子透過形式(pattern)配對的方式轉成函數，例如句子”蘋果有5顆”，透過系統會轉成(equal(蘋果數量)5顆)這樣的函數表達。然而這種方法有兩大問題，1)必須針對每種句子建如上的關係表達，但是真實的數學文字題目往往是比較自由且不太受限的，2)衡量結果之有效性存疑，因為那時候還沒有現在公開資料集的概念，也因此難以進行公平的評測。

之後陸陸續續有其他研究，到了2014年Hosseini[9]認為處理動詞才是數學文字題的根本，例如句子”John gave some kittens to Liz”，人們在解讀題目時因為看到”gave”這個動詞所以知道此句讓John's kittens原有量減少，並讓Liz's kittens的數量增加，於是Hosseini的ARIS系統即是根據以上的觀察，提出一個以動詞驅動狀態改變的解題系統，透過語法的分析辨別出相關的實體(entity)與其對應的值，以上述例子而言就是Liz's kitten和John's kitten和它們的數量，抽出動詞相關特徵，將其當作輸入餵進以SVM為基底的分類模型，預測出要執行的數學動作(加、減、乘、除)，然後進行計算，進而影響Liz's kitten跟John's kitten的狀態。

同年在ACL上Kushman[10]提出了一種統計學習的方法，名為KAZ8，引入模板的概念，針對每題標注一個數學式子，在訓練時每個問題抽取題目中不同層次的特徵，例如有關詞彙、詞性與語法等，經由模型預測題型，並將對應的值填近公式裡，例如題目為”小明有5顆蘋果，買了3顆，剩下幾顆”此題目的正確公式為 $x = a + b$ ，透過抽取題目中的特徵，期望模型能從很多公式模板中預測出對的算式，並且將5填入a，3填入b藉此算出答案。然而此方法有個缺點就是無法解決



訓練集之外的題型，比如訓練集只出現過兩個數相加，模型無法泛化解答三個數相加的問題，這顯然與我們想要的學習相差甚遠，也因此此類方法漸漸式微。

在神經網路崛起以後，給了數學文字題不一樣的解。2017騰訊AI Lab引進了在機器翻譯中有良好表現的sequence to sequence模型到數學文字題的領域，輸入即是題目敘述，經過一個sequence to sequence模型輸出符合此題目敘述的公式，還提出了一個相似度retrieval model，也就是現有一些有正確模板公式的題目，未知的題目以tf-idf的比對方式找出其最相近的題目，若相似度大於某個極值則直接套用模板。此外那篇論文中最重要的一點莫過於提出了一個公開的資料集Math23k讓後續的研究者能有一個評斷的標準。

2018年MathDQN[11]將Reinforcement learning相關技術應用至數學文字題，狀態跟行動分別為目前抽出的兩個數字以及運算符號(+,-,\*,/)。2019年Zhipeng Xie[12]以樹狀結構來處理數學文字題，根據每個階段的目標建出一個表達樹(expression tree)，如圖，同年Ting-Rui Chiang[13]以sequence to sequence加上stack的運用巧妙的處理此問題，實務上是在每個decode的時間點有四個步驟可選，分別是產生變數、將數字丟進stack、執行運算(pop stack中最上面兩個數字)、產生最後的結果。

儘管neural network-based的系統可以達到不錯的準確度，以上眾多解題系統都不會產生出解釋，導致我們很難對其模型做錯的題目進行錯誤處理，並進行改進，我們認為解釋可以直觀理解為一個系統了解題目語意與否的關鍵，如果能知道哪點算錯，我們就能對系統進行改進，但現今類神經網路還是黑盒子，無法直接地看到其決策的過程，這麼看來雖然我們最終提升了答題率，卻離可修改性漸行漸遠。

於是在2015年Keh-Yih Su[14][15]實驗室提出了一個對於數學文字題解釋生成不錯的解答，設計出一個從解題到解釋的一條龍的系統，作法是先對題目的每個句子做自定義的parse，將所有句子的資訊轉化成樹狀結構，之後透過一個SVM模型(特徵為題目的unigram、E-Hownet的語意特徵、名詞以及自定義的pattern match結果)來預測該做的運算類型，然後將預測出來的運算子與運算元搭成有結構的樹(父結點為運算子，子結點為運算元)。解釋部分就是針對每個運算子有設計幾個固定的模板，將運算元填入模板即可。



## 2.2.2 小學數學題解題系統

本篇論文提出了一個新的解題系統，命名為小學數學解題系統，我們的系統目標上與[14][15]類似都是要系統能解題又能生成解釋，不同的地方在於我們採用不同的方式來達成這個目的，首先我們將題目切分成數個子句，以句子為處理小學數學題的基本單位，透過觀察小學數學的資料集，將句子分為13大類，針對每一類問題以動詞為主軸，建立數個題目框架，以字義表達樹來儲存句子的資訊(semantic representation tree)。並且我們解題系統的架構是以自然語言去描述解題過程，這樣的好處在於可以讓各領域的專家撰寫解題流程，而避免程式設計師又要設計程式又要理解語言分析的情形發生。

此外為防止陷入早些年每題都必須設計框架，導致涵蓋率過低的問題，於是我們又設計了個”句子過濾”的機制，讓題目刪減到框架可以被系統對到的形式，藉此提升框架的通用性。並且在實驗的過程中，發現以”Label Sequence”的做法可以很好的結合我們的小學數學解題系統。因為這個發現本篇論文提出了一套簡易的流程將兩者進行融合，最終我們的系統除了可以算出以外，還會輸出依據框架定義的解釋，與前面眾多研究相比，我們的系統具有高度可修改性與可解釋性，為數學文字題開創新的解題思維。



# Chapter 3

## 資料集

本章節首先會進行問題描述，隨後介紹論文中使用的資料集。

### 3.1 問題描述與資料集

數學文字題即是給定一段題目敘述 $P$ ， $P$ 包含句子 $(s_1, s_2, \dots, s_n)$ 以及數字 $(v_1, v_2, \dots, v_n)$ 穿插於其中，我們希望可以求解公式 $E$ ， $E$ 由上述提及的數字組成，最後得到答案 $a$ 。在本篇論文中我們將問題進階為：除了求出答案跟式子以外，還要產生出生出此式子的解釋 $(e_1, e_2, \dots, e_n)$ 。

有鑑於現今其他多數研究是做在英文上，鮮少有繁體中文的資料集，於是我們從三個出版社康軒、翰林以及部編收集各年級的數學題目，匯集成小學數學資料集，各出版社題型分佈總結成Table3.1。

Table 3.1: 各出版社題型分佈

|    | 基本乘除 | 基本加減 | 進階乘除 | 進階加減 | 混和算式 | 概念題 |
|----|------|------|------|------|------|-----|
| 康軒 | 654  | 427  | 359  | 212  | 365  | 29  |
| 翰林 | 793  | 444  | 383  | 283  | 703  | 51  |
| 部編 | 128  | 22   | 91   | 35   | 89   | 6   |

分類中，基本題的定義是指那些依照題目敘述，只要列一個式子即可算出答案的題目，進階題則是要兩個句子以上或是做單位換算的題目，混合算式則是解題的公式牽涉到加減乘除，概念題通常考一些數理邏輯，而不會牽扯到數學式子的計算，底下Table3.2是各類型題目的例子。



Table 3.2: 各題型例子

| 題型   | 句子                                   | 答案             |
|------|--------------------------------------|----------------|
| 基本加減 | 一盒草莓有36顆，媽媽吃了15顆後，還剩下幾顆              | 21顆            |
| 基本乘除 | 一盒彩色筆有24枝，2盒彩色筆有幾枝                   | 48枝            |
| 進階加減 | 小華有2054元，小英有1969元，小慧有388元，三人共有幾元     | 4411元          |
| 進階乘除 | 工廠將3000枝鉛筆平分成50箱，再將一箱鉛筆平分成5盒，一盒鉛筆有幾枝 | 12             |
| 混和算式 | 一盒蛋捲有5根，阿楚買3盒，吃掉1盒，還剩下幾根             | 10根            |
| 概念題  | 11到20的數中，哪些數是奇數                      | 11,13,15,17,19 |



# Chapter 4

## 小學數學解題系統

本章節會介紹我們的數學文字解題系統。

### 4.1 小學數學解題流程

小學數學解題系統是以句子為單位，一個題目的句子進來以後，先依照標點符號切出幾個子句子，然後對句子做簡化，生出框架樹，將省略的詞補進去，最後跑解題程式生出解釋，整個流程如圖4.1

以圖4.1的例子為例，在reduction步驟時將3顆以特殊星號做刪減，將句子簡化為吃了某物，如此一來框架就能寫的比較簡單與通用。然後生出框架樹，在此稱作instance map，並將”吃了3顆”跟”剩下幾顆”句子中缺的”蘋果”補上去，有了instance map這樣的樹狀結構我們就能去跑解題的程式，依照解題過程生出相對應的解釋，底下我們會針對各步驟深入介紹。

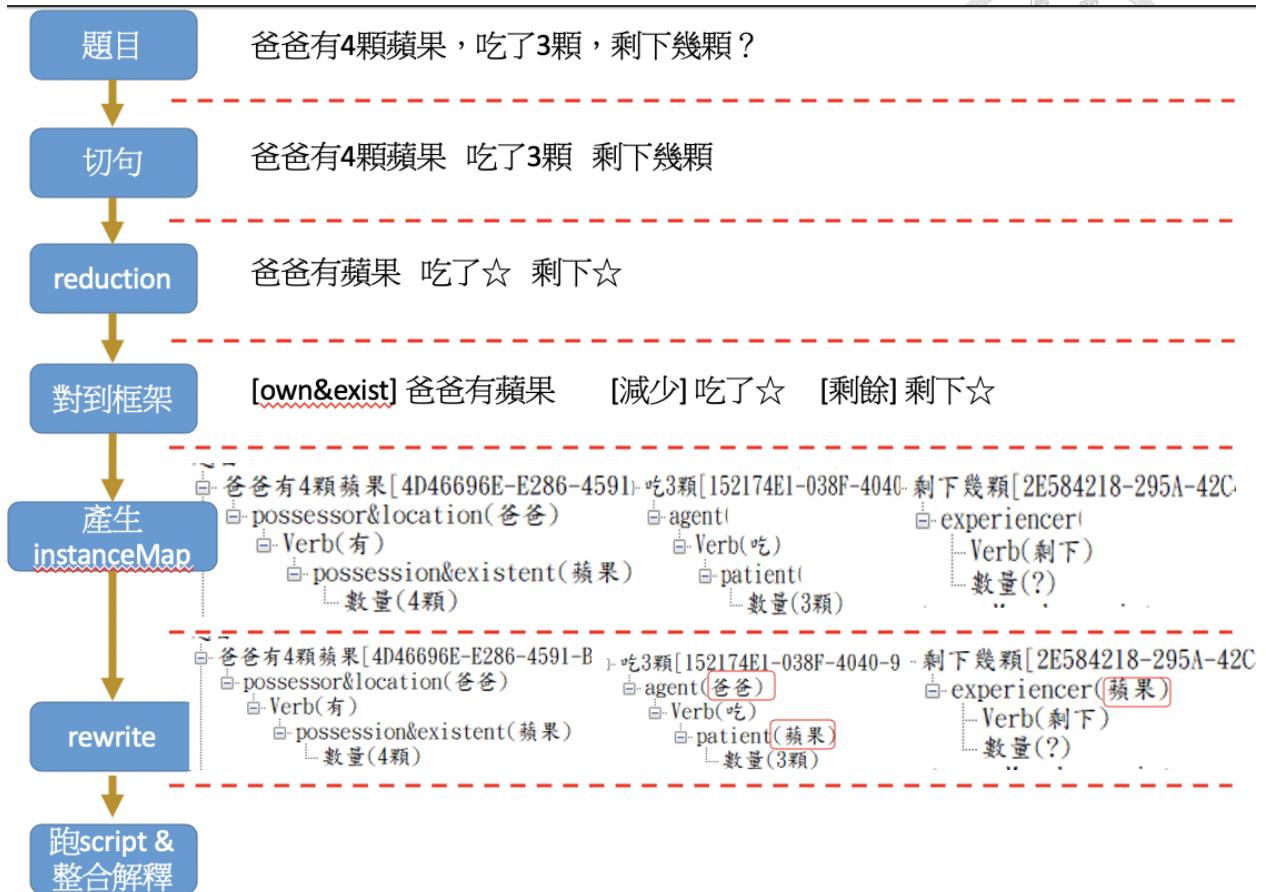


圖 4.1: 小學數學系統解題流程圖

### 4.1.1 切句與簡化(reduction)

小學數學解題系統是以句子為對到框架的基本單位，會有這樣的考量是因為若以句子為單位去做設計能盡量減少所需要寫的規則，以底下的例子而言，兩個題目明明很相近，假若是每個題目都去設計一個解題流程，就無法抓到這樣的關係，並且也會因為句子有很多種可能，導致必須窮舉無數個解題流程，也因此以句子為單位是較為合理的做法。

實作上也相當簡單，就是以標點符號作為斷句子的依據。

- 1. 爸爸有4顆蘋果，吃了3顆，現在有幾顆蘋果？

答案 = 4 - 3

2. 爸爸有4顆蘋果，買了3顆，現在有幾顆蘋果？

答案 =  $4 + 3$



有了各個子句後，我們會對句子進行簡化，這步驟的目的在於簡化句子的架構，讓句子簡化到盡可能留下最基本的主詞、動詞、受詞的架構，如此一來我們也不用設計太多的句子框架。

為了具體化這樣的想法，我們會有個簡化的函式庫，如圖4.2，每個簡化的函數底下會有幾個元件，HAS-Part、INSTANCE、ontology以及Role，HAS-Part的功能是列舉要被這個框架對到的句子應該要有哪些詞，INSTANCE是這些詞應該以怎樣的排列才能對到這個框架，ontology告訴我們對到此框架的句子該生成怎樣的表達樹，最後Role是把在HAS-Part中而ontology定義的表達樹會用到的關鍵詞特別拿出來，並且可以做別名。

以圖4.2中的簡化法架構圖跟句子”3位小朋友”做解釋的話，HAS-Part底下的Head代表會被留下的詞，所以會對到”小朋友”，Reduction表示會被縮減至Head的樹狀結構底下的詞，也就是3位。而例子中的這句話可能有兩種排列組合，”3位小朋友”和”小朋友3位”，這兩種說法都適用於此簡化規則，所以INSTANCE列出兩種組合。ontology因為要被簡化的詞已經被縮到Head底下，故只有Head這個節點。由於在此簡化規則中Reduction跟Head缺一不可，因此兩個點都包含在Role底下，最後原始句子就會被簡化成小朋友。

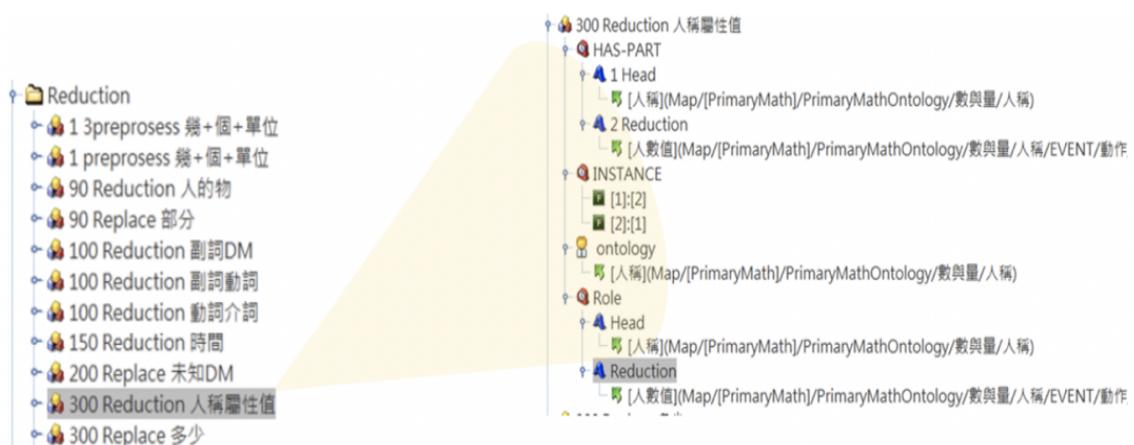


圖 4.2: Reduction機制圖



#### 4.1.2 對到框架並產生instancemap

對於每個句子而言，我們認為用樹狀結構儲存句子所包含的資訊是最直觀也最容易執行的，於是在小學數學解題系統中我們依照對小學數學句子做的觀察，大致上將句子分成13種類型，針對每個類型我們設計如圖4.3的架構，”屬性”就是13種中的其中一種句子類型，符合屬性的句子通常是描述物件的數量，例如麵粉4435公斤和142個甜甜圈都屬於此框架，以圖中為例，框架底下各個節點的功能與簡化法如出一轍，不同的地方在於在簡化法中的ontology我們改成instance map來做區別，以及多了ScriptNL、ScriptNLTransform，這兩個點後續在4.1.4會多著墨。



圖 4.3: 小學數學系統中的框架架構圖

所以整個流程圖如4.4，句子進來以後會判斷各個字詞屬於”HAS-PART”中的哪個類別；轉成類別以後，判斷這樣的類別序列是否存在我們定義的”INSTANCE”中，依照結果生出相對應的”InstanceMap”，這樣子就能將句子的資訊存在樹狀結構中。

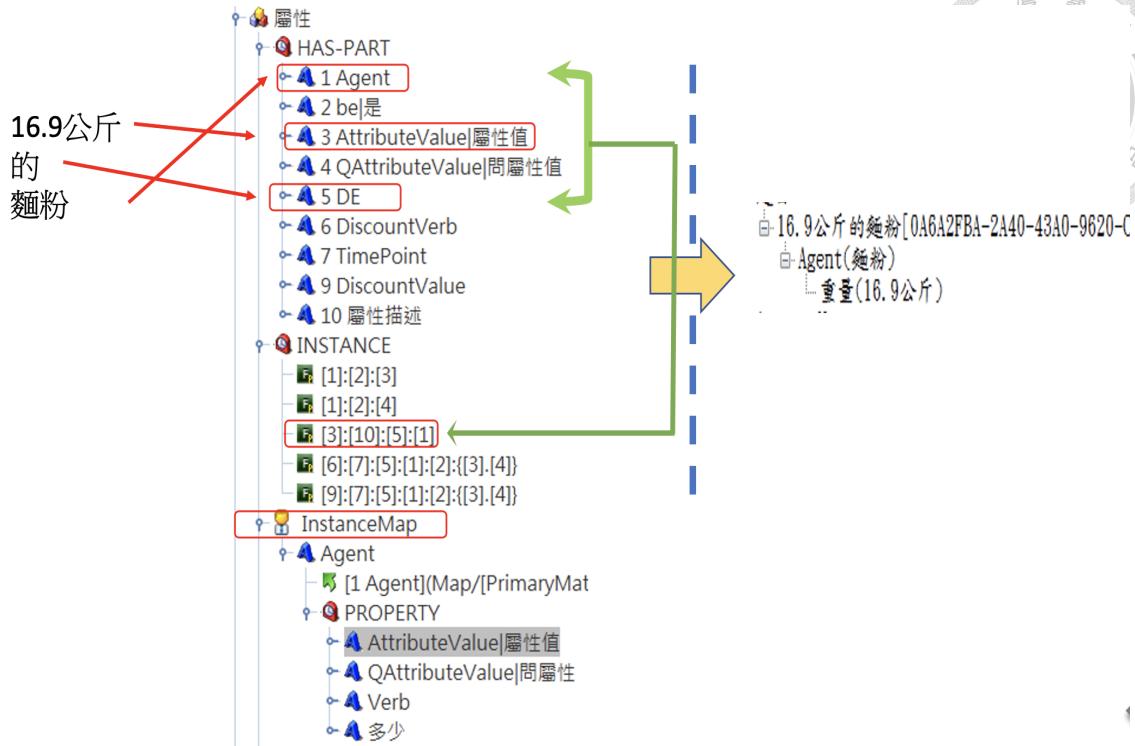


圖 4.4: 句子產生instance map的流程圖

### 4.1.3 改寫(rewrite)

在經過前述步驟後，我們已經有各個句子的表達樹了，然而還有些被省略的受詞需要填入，以底下的例子一而言，”爸爸有4顆蘋果，爸爸吃了3顆，剩下幾顆？”，在第二句跟第三句中的蘋果皆被省略了，所以我們透過名量詞的配對，將”蘋果”填入。而例子二是如果省略的是主詞並且受詞也被省略的話，因為通常這樣的題目是只有單主詞的，所以我們會找最近的主詞填入。例子三是省略主詞且有多個人稱時的情況，這時我們會透過主詞跟受詞的配對找出應當填入的主詞。經過填入這些被省略的詞後，最後就能生成完整的表達樹，至此我們已經將題目中各個句子的重要資訊提取出來，接著就能透過script來生成解題流程。

- 改寫例子

1. 爸爸有4顆蘋果，爸爸吃了3顆，現在有幾顆蘋果？

改寫結果 = 爸爸有4顆蘋果，爸爸吃了3顆蘋果，現在有幾顆蘋果？



2. 爸爸有4顆蘋果，吃了3顆，現在有幾顆蘋果？

改寫結果 = 爸爸有4顆蘋果，**爸爸**吃了3顆蘋果，現在有幾顆蘋果？

3. 小明有4顆蘋果，小華有4根香蕉，吃了3顆蘋果，現在有幾顆蘋果？

改寫結果 = 小明有4顆蘋果，小華有4根香蕉，**小明**吃了3顆蘋果，現在有幾顆蘋果？

#### 4.1.4 跑script與整合解釋

在做解題流程時，我們引入了RNL(Restricted Natural Language)，此機制的功用是讓我們可以以自然語言去描述解題過程，並且restricted的意義在於我們希望撰寫的自然語言描述能儘可能的簡單易懂，這樣也比較容易轉譯成程式，具體上是像圖4.3中有節點ScriptNL以及ScriptNLTransform，ScriptNL節點底下存的是當遇到此題型時該怎麼做處理的模板，也就是自然語言描述解題過程的部分，之後ScriptNLTransform將這些模板轉換成程式可處理的函數，透過這樣的方式讓其他人也能進行ScriptNL的編寫，且撰寫者可以是有相關領域知識的人，這樣設計出來的ScriptNL會更加穩定與通用，並且程式設計師能專注在後端程式函數庫的開發。

例如題目”爸爸有7顆蘋果，吃了2顆蘋果，剩下幾顆？”，由於小學數學解題系統是以句子為單位做處理的，所以當看到句子”吃了2顆蘋果”時，我們會想找前面是否有提到蘋果的數量，然後加其減2，ScriptNL就是將這樣的一個流程寫下，要注意的是雖然我們是以流程圖的概念去設計ScriptNL，但其實在實作上ScriptNL是以節點的方式呈現，如圖4.5所示。

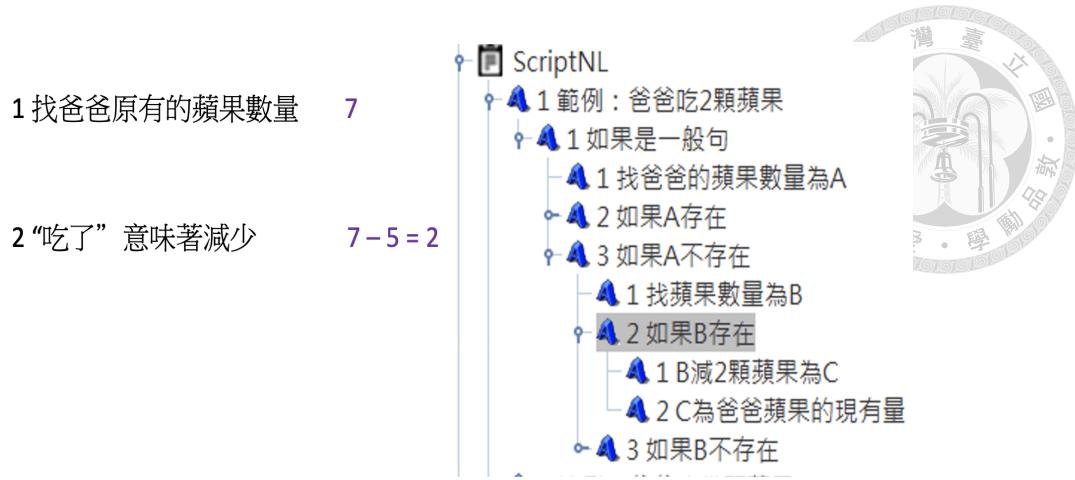


圖 4.5: ScriptNL例子

然後ScriptNLTransform流程如圖4.6，我們會建一個函數庫，這個函數庫的作用在於將ScriptNL轉化成一系列程式可以看的懂的函數，以底下的圖”如果B存在”為例，如果會被”如果”的節點對到，B會被”英文字”這個節點對到，存在會被”存在表示”對到，而Condition底下代表的是轉化後的結果，所以經過轉化後就會變成”if(isfire(B))”的結果；並且對於函數庫中缺少的解題過程只需要由解題過程的編寫者告知需求給程式設計師去做開發即可，如此一來各司其職，能讓整個系統更加的系統化與穩定。



圖 4.6: RNL轉化示意圖



解釋生成的部分，我們是對每個解題流程中設計一套範例解釋，當有新的句子進來時就對範例解釋裡面的字詞進行代換，以下例子為例，黑字以外都是可以被代換的字，用不同顏色區隔代表的是不同的詞。以這句範例解釋而言，吃了、蘋果以及2顆是可以被代換的詞，所以當相同框架的句子依照此範例生成解釋時，只需要將範例解釋中的詞替換為與之相對應的字詞即可。

- 解釋代換示意

1. 爸爸吃了2顆蘋果

範例解釋：吃了是「減少」的概念，所以蘋果原有量-2顆

2. 媽媽借走4張貼紙

最終生出的解釋：借走是「減少」的概念，所以貼紙原有量-4張

值得一提的是我們將解題流程設計為節點組成的樹的好處在於當不同的題目進來以後，若走的是不同的路徑，產生出來的解釋也不同，間接達到解釋動態生成的概念。

透過以上的流程小學數學解題系統就能根據算出的答案生出相應的解釋，而這樣的系統是具有高度可修改性與可解釋性的。



# Chapter 5

## 句子過濾

本章節會介紹數個符合小學數學解題系統的句子過濾的方法

### 5.1 通用過濾法

在上一章節中我們介紹了小學數學解題系統基本的運作流程，由於小學數學解題系統是以句子的框架為基礎去設計的，所以我們需要的是一個通用的框架，不然很容易流於窮舉各個句子簡化規則的情況。

儘管在小學數學解題系統中已經有根據小學數學解題系統客製化的簡化機制，但是小學數學文字題的敘述是更加自由且不受限的，也因此有些時候句子中有非常多對於框架無關的冗言贅字，例如底下的例子，我們也很難透過窮舉的方式全部列舉於小學數學解題系統的簡化機制中，並且當簡化機制一多時，使用各個簡化法的順序也就顯得很重要，也因此透過一個過濾機制，讓句子盡可能的簡單化是現階段一個可以改進系統的替代方案。

- 需要簡化的例子

1. 但是根據客戶要求需趕在12天內完工

精簡過後的例子 = 客戶要求12天內完工

故本章節引入了幾個額外的通用過濾機制，期望透過這樣的方式處理一些零碎的簡化情況，讓小學數學解題系統中的簡化機制只要著重於系統導向。



### 5.1.1 Rouge-N based

框架也就是一個句子中隱藏的句法結構，也可以視為詞性的組合，如何只提取中句子中符合系統定義的框架有一個簡單的想法是把目前小學數學解題系統可以處理的句子都轉成一個詞性的序列，建成一個字典，我在此採用的中研院開發且開源的NER套件”ckiptagger”[16]，此系統利用自注意機制(self-attention)與Cross-BiLSTM模型做到高準確度的中文詞性標註，因此非常適合我們的任務。

於是乎整個流程為新的句子進來後，透過ckiptagger將其轉成詞性序列後，再與現有的字典中的所有句子的詞性序列進行比對，選出一個最相似的句子，然後盡可能的改寫成與相似句一樣的句構。

比對的算法在此是採用Rouge-N，Rouge-N是用來評比兩個句子相似程度的一種算法，一般用在摘要的比對上，公式如5.1， $m$ 跟 $n$ 分別代表字串 $X$ 跟字串 $Y$ 的長度，LCS(longest common subsequence)是最長共同子序列，由於目標是讓字串 $X$ 跟字串 $Y$ 盡可能地相近，故在此我只採用recall來當比較依據，而非常用的F score。

找到最相近的句子後，改寫依據為兩者的LCS，僅留下其LCS的子字串，其餘皆省略。

$$\begin{aligned} R_n &= \frac{LCS(X, Y)}{m} \\ P_n &= \frac{LCS(X, Y)}{n} \\ F_n &= \frac{(1 + \beta^2)R_nP_n}{R_n + \beta^2P_n} \end{aligned} \quad (5.1)$$

### 5.1.2 Transformer based

在通用過濾機制中我們希望設計出的額外過濾機制可以讓所有新句子經過一層字詞的刪減轉變成可以被小學數學解題系統處理的句子，因此可以把面臨的問題歸化成字詞的分類問題。也就是一個句子進來希望透過某個決定的機制決定出要被留下的字詞，要能設計出這樣的機制必須了解小學數學解題系統可以處理的句子特徵有哪些，而類神經網路恰好非常善於提取輸入的特徵。所以我們可以設計一個深度網路模型，當有輸入句子時，模型會輸出一串0與1的序列，這個輸出序列與輸入句子長度相符，0代表此字詞應該被省略，1則要保留，透過大量的資料訓練，可以模擬出一個穩定刪減模型。

然而有個問題是我並沒有訓練資料，也就是句子與此句子經過修改後可被小學數學解題系統對到的形式的配對，有的只是在6300題小學數學文字題中，哪些



題可以正確算出答案哪些不行的間接性資料，於是一個簡單粗暴的想法是取那些可以算出正確答案的題目，對這些題目中的每個句子做增字，多加進去的字則代表應該被刪去，標註為0，原本在題目中的字詞則標註為1。

然而這樣的加字原則會產生一個問題，如果只是隨意亂插字，那插入的字在原本句子中就會顯得很突兀，這樣深度網路模型可以透過簡單的詞向量分辨出哪些是增加的字，而不是真正學習到刪減的機制。一個折衷的辦法是將所有句子都轉成詞性序列，改成增加詞性進去，如圖5.1。

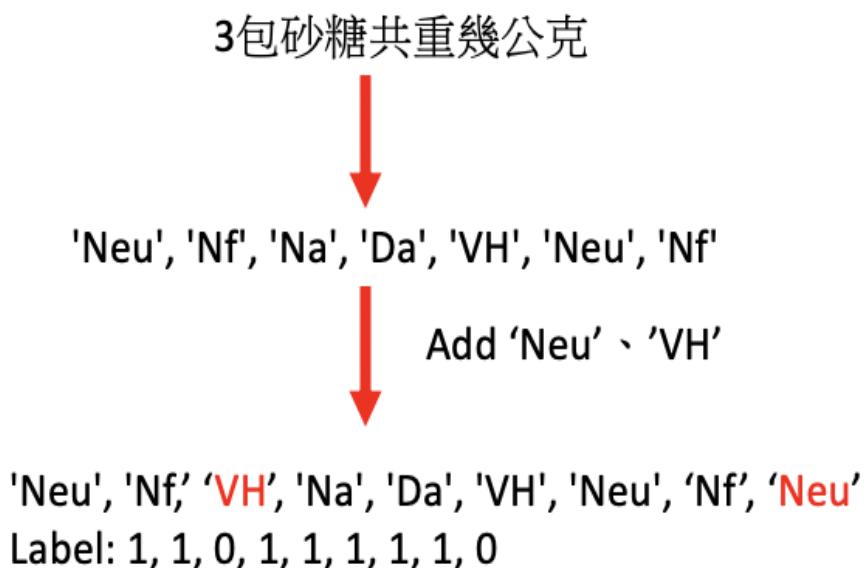


圖 5.1: 詞性增減示意圖

為求盡量達到測試與訓練資料的一致性，我假定每個算不出答案的題目中的句子轉成詞性序列後，都可以透過LCS找到與其最相近且可被小學數學解題系統處理的句子的詞性序列，統計兩者之間的字數差距分佈，當作插入字數的依據，分布如圖5.2。此外為了避免連續兩個相同的詞性在一塊，但因為加的位置不同，導致資料真實標註不一致的情況發生，於是會在生資料時做額外的判斷。

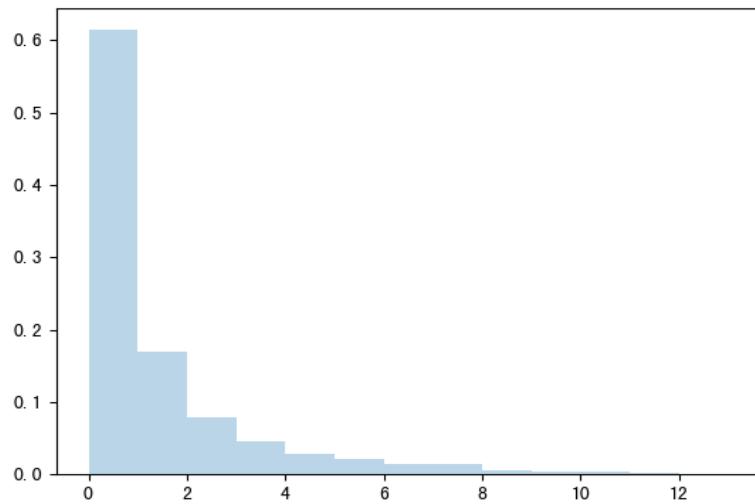


圖 5.2: 刪減字數分佈圖，縱軸是佔的比率，橫軸是刪減的字數

由於輸入是包含時間資訊的詞性序列，因此在模型的選擇上以可以處理時間資訊的RNN、LSTM及Transformer為主。又因為許多研究顯示[2][3][17]，Transformer的表現較前兩者好，所以模型我以Transformer當主幹後面接一層線性層5.3。

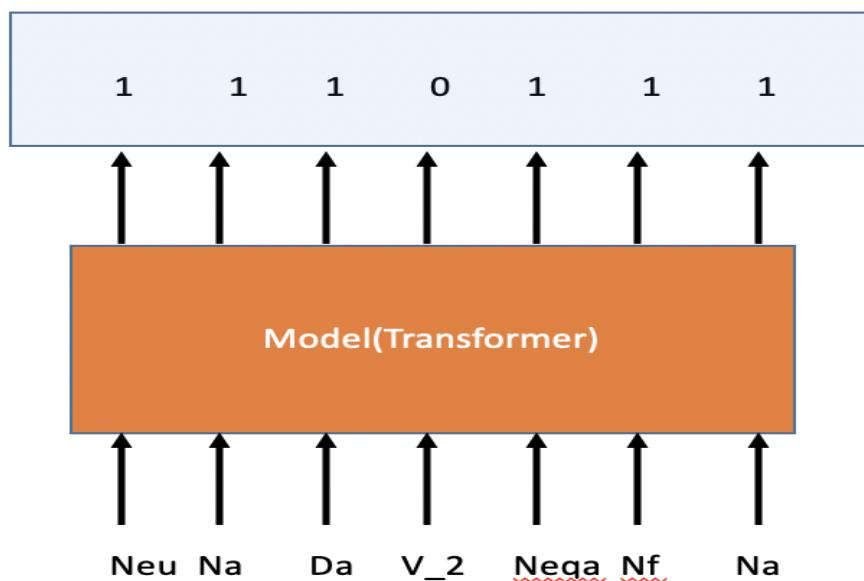


圖 5.3: 模型架構示意圖

整個訓練流程就是給定輸入詞性序列 $(x_1, x_2, \dots, x_n)$ ，模型輸出過sigmoid層，要預測出概率 $(y_1, y_2, \dots, y_n)$ ，損失函數是去計算與真實標註 $(\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_n)$ 之間



的二元交叉熵(binary cross entropy)，由於在訓練資料中要留的字遠多於要被丟棄的，所以爲避免模型最後都選擇留下字，在算損失時對於選擇要留下的字必須多乘上一個權重，整個式子如5.2。

$$-\sum_{k=1}^n [w \cdot \tilde{y}_k \cdot \log y_k + (1 - \tilde{y}_k) \cdot \log(1 - y_k)] \quad (5.2)$$

### 5.1.3 Pre-trained BERT based

以上兩種作法完全不考慮字義只著重於句法，這是一個在沒有訓練資料下的妥協作法，好處是模型可以專注於在對到框架，但是如果只考慮框架不考慮語意，原始句子可能會因爲刪減而導致不通順，儘管這樣的句子可以被小學數學解題系統對到且算出答案，但是由於句子是殘缺的導致小學數學解題系統生出來解釋也會是不完整的。

幸好根據觀察，小學數學解題系統的框架只取一個句子裡面最關鍵的部分，這樣原生性的特質恰好可以讓我們把原本不知道要刪減哪些字的無標註問題，轉化成刪減一個句子中的冗言贅字，也就是要設計一個神經網路模型，這個模型輸入句子後能知道哪些字詞是較不重要的。於是受到[18]的啓發，一個句子中較不重要的字詞刪減後，對於其終端任務的影響是較小的，舉例而言若是在做句子分類問題時，最不重要的詞肯定是對於最後模型預測分類的分佈影響最小者，將這樣的觀察以數學式子表達的話則如5.3， $f(y|x)$ 是給定句子 $x = (x_1, x_2, \dots, x_n)$ 後模型預測出標註 $y$ 的機率，而 $y = \text{argmax}_{y'} f(y'|x)$ 也就是原始模型會預測出的標註， $g(x_i|x)$ 是我們設計的句子過濾模型。

$$g(x_i|x) = \text{argmin}_i(f(x) - f(x_{-i})) \quad (5.3)$$

在論文[18]並不是直接去執行上述公式，原因在於若句子長度爲N，則對於每個句子模型就需要跑N次，當模型參數量很多時在運算上會耗時相當多時間，這顯然是一個不太可行的方法，所以論文中採用一個簡化的思路，就是去算句子中的每個字的詞向量對於最後預測結果的梯度的影響，實際上是以詞向量內積梯度，這樣每一輪刪減字時就只需要去進行一次的額外運算即可(因爲乘法可以平行運算)，透過這樣的簡化方式去擬合式子5.3的概念。這樣的問題轉換在數學上也是合理的，這個問題的經過[19]背書的，在那篇論文中對於這樣的轉換過程有進行詳細的數學推導，並且也有多篇論文[20][21]採用這樣的方式。

儘管這樣在小學數學的資料集上仍有個問題，就是沒有訓練資料何來的梯度呢？於是問題又退回直接求解5.3，但同樣的問題又延續了過來，沒有模型 $f$ 所以



也難以直接去計算每個字詞的重要程度，那何不再退一步思考，問題在於找到一個字詞對於句子最不重要，一個想法是如果句子可以表示為一個向量的話，刪減掉的字只要能最小化與原句子間的向量差異即可。剛好在章節2中提到了很多模型善於將句子轉化為一個K維的向量，我取其中提及的BERT當作用來計算句子詞向量的模型，選擇的原因在於BERT有針對中文語料去做訓練且開源的預訓練模型，並且跟XLNet比起來，BERT在我們下游的小學數學文字題取得了較好的成效。

在原始的BERT中對於中文字詞做Mask時是以字為單位，而非以詞為單位，但是在中文裡有些字單獨是沒有意義的，例如“徘徊”這詞單獨的“徘”或“徊”都無法各自成義。所以針對中文語料在做遮罩時，當遮到某個詞裡的某個字時，必須把那個詞中剩餘的字也遮住才較為合理，全詞遮罩示意如圖表5.1.3。

| 說明       | 舉例                               |
|----------|----------------------------------|
| 原始文本     | 我徘徊在無人的街頭                        |
| 分詞文本     | 我徘徊在無人的街頭                        |
| 原始Mask輸入 | 我[Mask] 徘徊在無人的街[Mask]            |
| 全詞Mask輸入 | 我[Mask] [Mask] 在無人的[Mask] [Mask] |

因此實作上我不是直接使用HuggingFace開源裡的BERT，而是採用[\[22\]](#)所改進的RoBERTa-like的BERT，RoBERTa-like的BERT與原始的BERT有四點不同1)在預訓練的階段採用全詞遮罩，而非BERT以字遮罩，理由如上所述，並且不像RoBERTa一樣使用動態遮罩;2)取消了在BERT預訓練時的跛具爭議性的下句預測(Next Sentence Prediction);3)將輸入句子字數的限制從128提升到512，這樣的改動可以讓模型更好的學到長距離依賴(long dependency);4)訓練步數適當的延長。透過這樣的改動可以讓模型做到比原始的Bert更好的表現。

模型選定以後，整個流程也就明朗化，給定句子 $X = (x_1, x_2, \dots, x_n)$ ，先經過ckiptagger斷詞後得到斷詞後的結果 $W = (w_1, w_2, \dots, w_k)$ ，將刪掉第*i*個字詞的句子 $W_{-i}$ 通過RoBERTa-like BERT模型得到每個字詞的隱藏向量 $H$ ，將 $H$ 取平均後當作這個句子的句子向量，這裡不取第一個字("[CLS]")的隱藏層向量當作整個句子的句子向量，原因是因為在Hugging face提供的BERTModel有說第一個字通常無法代表整個句子的意思，建議是取平均來當作句子的句子向量，並且經由我做實驗發現取平均的表現略高於取第一個字"[CLS]"的結果。然後計算每次刪掉一個字詞後的句子向量與上個階段刪完字後的結果的句子的句子向量的歐式距離，若小於某個閾值則代表此字詞是可被忽略的詞。



儘管這樣的改動導致每個句子(假設句子長度有N個字)必須跑模型N次，但由於在小學數學問題中句子本身就比較短，再加上斷過詞，所以就算每次遮蔽一個字詞就去跑模型一次，也不至於花太久的運算時間。此外為了提升刪字的準確度，我還利用Beam search的機制，每一輪將所有可以被忽略的字加進候選人列表裡，以刪減後的句子與原始句子的句子向量的歐式距離做排序，取beam size大小的句子到下一輪，直到後選人列表為空，整個演算法如1。

---

**Algorithm 1:** RoBERTa-like BERT deletion

**Input:** A Set of sentences in all intractable question F, RoBERTa-like model M,

threshold t

**Output:** R is amendment of F

```
1 R = [];
2 B is searching list;
3 for all sentence s in F do
4     h = M(s).mean();
5     B.append(s), P=B;
6     while B is not empty do
7         bs = P.pop();
8         bsh = M(bs).mean();
9         ss are those sentence which are removed one word from bs and the distance
          between sentence embeddings of each sentences and bsh is lower than t;
10        B.append(ss[:k]);
11    end
12    R.append(P[0]);
13 end
14 return R;
```

---

如果僅照上面的演算法1會刪掉一些對於句子來說不是很重要但是對數學文字題相當重要的資訊，以底下的例子來看

- 我們創建的句子集合實例

1. 6條吐司平分給15個人，和12吐司平分幾個人時，每個人分到吐司一樣多？

被刪除的字 = 條



2. 一盒蘋果汁375毫升，一盒木瓜牛奶500毫升，7盒蘋果汁和1盒木瓜牛  
奶合起來共有幾公升幾毫升？

被刪除的字 = 一、盒、共、有、幾公升

3. 小芳每天存5元，星期可以存多少？

被刪除的字 = 一

可以發現多數情況下，單位跟數量被當成優先刪除的選擇，然而這點與小學數學解題系統的框架是背道而馳的，所以我會在做遮罩時額外做判斷，讓模型不去計算數字與單位被遮罩後的句子，並且由於小學數學解題系統是以動詞為主要框架，因此動詞被遮罩的結果也不會被列入考量，透過這些額外的限制，可以讓模型的結果更加貼合小學數學解題系統所要的格式。

BERT based的演算法1也能從小學數學現有的訓練資料(哪些題是可以被小學數學解題系統對到的，哪些不行)上得利，具體算法是先將所有可以被小學數學解題系統辨識的句子通過ckiptagger轉成詞性序列，建成一個字典，每次改寫完後轉成詞性序列去比對字典，若詞性序列相符則視為句子已經被裁剪到小學數學解題系統可以對到的句子，就不再進行修改，反之則繼續裁剪，這樣就能與現有的訓練資料做結合。

#### 5.1.4 Label Sequence

另外一種從字義出發的思維是當一個句子進來時，只取我們在乎的詞語，將這些詞語重組成一個句子，透過這樣的方式可以從另一個角度達到過濾的效果。

實作上是建一個字典，裡面包含數種類別，每個類別底下是我們所在乎的符合此類別的字詞，我們稱這樣的做法為”Label Sequence”。圖5.4為此系統的流程圖。

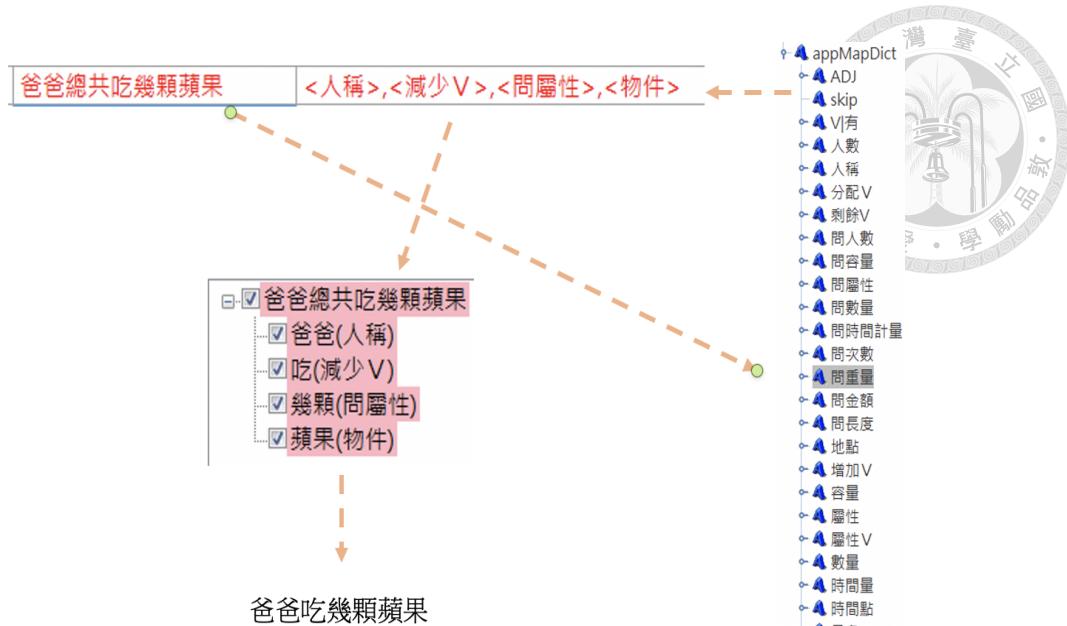


圖 5.4: Label Sequence 流程圖

Label Sequence還有個的功用在於找相似句，這是因為人類在遇到問題時，往往也是去找以往學過的相似問題來當作模板，將解法複製過去，於是Label Sequence的設計就可以讓我們找到每個句子的相似句型，然後針對此句型設計其instance map，以此取代過去以動詞為框架的概念。然而在小學數學解題系統中，解題流程依然是建立在舊的動詞框架的instance map底下，比起將解題流程全部重新編寫，將兩個instance map融合起來顯然更簡單點，因此如何將兩者做結合也成為一個相當重要的課題。

更由於Label Sequence建的字典基本上是取自於小學數學解題系統中，也因此彼此之間是高度相關性的，話句話說就是原本小學數學解題系統可處理的題目就算先跑過Label Sequence做刪減往往也不會有所影響，這點在後續的表格6.2.3也得到應證。又因為Label Sequence的可修改性，於是此方法是最切合我們的系統也非常適合當作一個先行的通用過濾程式。

針對此問題我提出了一個簡易的流程圖5.5。首先當一個句子進來時，會將其轉成Label Sequence，用一個分類器預測此Label Sequence應該屬於小學數學解題系統中的哪個動詞框架，取出動詞框架的instance map，將對應值填進去即可。又因為我們希望使用者可以針對Label Sequence設計出一個符合此Label Sequence的樹狀結構，所以在實務上做的其實是將客製化的樹與舊有的instance map做結合，底下會個別敘述各個步驟的實作。

分類器的部分，我其實是建一個字典，裡面的key值跟value分別是label

sequence與對應的框架，表5.1.4是我針對目前可以處理的所有題目的句子依照9:1的比例切成訓練集與驗證集進行測試的結果，為了盡量弭平切分驗證與訓練集所造成的差距，我切100次的結果去進行平均，可以發現以這樣建字典的方式幾乎可以達到百分之九十九的正確率，而且在總句子3500句中，不同label sequence的種類透過一些過濾機制也能減少到只有365種，因此其實只要設計得夠好，label sequence跟框架的之間可以達到接近於完美的對應並且框架數也不會到過於誇張。

|      | 正確率  | 框架數 |
|------|------|-----|
| 訓練結果 | 99.1 | 365 |

在融合樹的方面我以觀察發現舊有小學數學解題系統中的單位數會對應到Label Sequence的”數量”、而單位量會對應到”屬性”、Agent相等於”物件”，透過這樣的關係，以BFS搜尋的方式遍歷依照Label Sequence客製化出來的樹，將值依序填入即可。

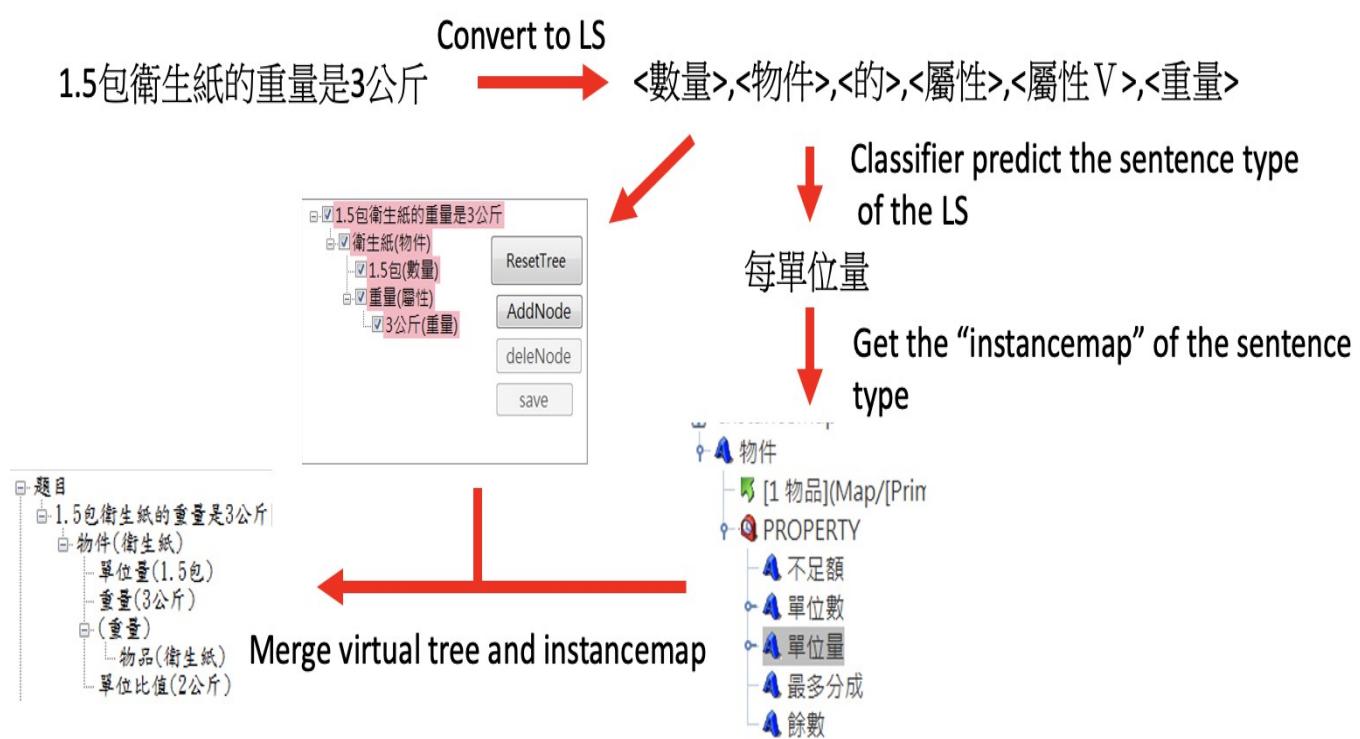


圖 5.5: Label Sequence 加上小學數學解題系統流程圖



# Chapter 6

## 實驗

本章節包含研究相關數據，由於RNL系統跟句子簡化差異甚大，故各別做實驗來呈現。

### 6.1 小學數學解題系統答題統計及分析

我們使用小學數學解題系統處理在章節3介紹的小學數學資料集，答題狀況如表6.1，在6000多題中小學數學解題系統答對 $\frac{1}{6}$ 的題目，並且對於那些算對答案的題目RNL將近9成以上都能生出正確的解釋。

|          | 總題數  | 答對題數 | 解釋正確且答對題數 |
|----------|------|------|-----------|
| 小學數學解題系統 | 6032 | 1096 | 1021      |

為了詳細瞭解答題情況，於是畫出答對與答錯題目的題型分佈，圖6.1顯示了在概念題與混和算式上是兩者差距最大的題型種類，分析原因可能是因為概念題一般都是考一些數學觀念，所以如果觀念尚未被建構的話小學數學解題系統就無法處理，而混和算式則是通常有一段冗長的題目敘述以及需要較複雜的計算式子，我們的系統的簡化機制可能無法從中正確的抓出重要資訊，亦或是解題流程還未編寫，也因此導致這兩種題型分佈差距較大。

由於小學數學解題系統是先對中框架，然後跑解題流程，解題流程的部分又牽涉到後頭程式的實作，所以答錯的題目有可能有四種錯誤:1)框架尚未建立，2)程式後端錯誤，3)解題流程尚未編寫或有誤，4)對錯框架。第四種是最特別也最難以估計的，因為目前小學數學解題系統的機制是當句子未對到框架時則會不斷進行簡化，然而若是某個正確句子要對到其正確框架是必須經過簡化2然後簡

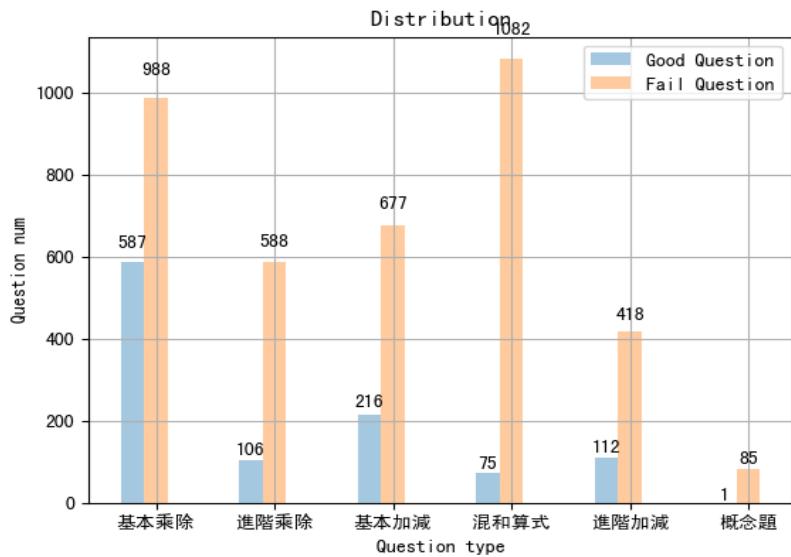


圖 6.1: 答對與答錯在題型上的分佈

化1，但以目前小學數學解題系統是一律以編號小的框架先執行，像是這樣系統上設計的問題可能就會產生第四種錯誤，並且現階段尚未有一個程式能正確的辨識輸入的句子應該屬於哪個RNL框架，所以在表6.1中就不列出第四種錯誤，但是要注意的是在一般狀況下，第2種跟3種錯誤是有可能包含第4種錯誤的。

|    | 框架尚未建立 | 後端程式錯誤 | 解題流程尚未編寫或有誤 |
|----|--------|--------|-------------|
| 題數 | 1170   | 126    | 3634        |

## 6.2 各簡化機制實驗

### 6.2.1 額外答對題數比較

由於目前無法知道哪些題目經過簡化機制後可以被小學數學解題系統正確的算出答案，因此為了公正性，我們假定所有題目皆可以經過修改，轉變成可處理的題目，於是將這4935題先經過各個簡化法後得到改寫的題目，再將此題目交由小學數學解題系統做運算。表4.2列出了採用各個方法後，能額外答對的題數。

在Transformer based的做法中，我學習率起始為0.01，每當驗證資料集的損失上升時，則減半學習率，直到小於 $10^{-6}$ 為止，詞性向量維度為40，Dropout率設



|      | Rouge-N | Label Sequence | Transformer | BERT |
|------|---------|----------------|-------------|------|
| 答對題數 | 76      | 160            | 126         | 209  |

Table 6.1: 各個簡化方法的比較,I Rouge-N代表迭代Rouge-N算法

為0.2，Transformer head數為4，層數3層，輸出的機率大於0.7時才視為應該保留的字詞。而BERT based的做法中距離閥值為2.86。

可以發現NN based的做法答對題數比非NN based的做法高出一些，主因可能得利於龐大的訓練資料，Transformer based的訓練資料有50000筆，驗證資料5000筆。而BERT based更是訓練在龐大的維基語料上，相比之下Rouge-N方法的訓練資料僅有1000多題答對的題目，以及其子句，也因此後兩者在表現上勝過Rouge-N based也是情有可原的。儘管有龐大的訓練資料，Label Sequence(LS)的表現依然勝過Transformer based，原因可能得利於LS作法與我們的系統的高度契合性導致，因為我們在設計LS作法時所挑選的字就是我們系統所需要的字，也因此LS改寫完後的句子通常都是能被小學數學解題系統所辨識的。

|   |  |
|---|--|
| 原句：翰翰有39張色紙，小誠有77張色紙，兩個人一共有幾張色紙?              |  |
| R   | 翰翰有39張色紙，小誠有77張色紙，兩個人一共有幾張色紙?                                |
| LS  | 翰翰有39張色紙，小誠有77張色紙，兩個人 <b>一共有</b> 幾張色紙?                       |
| T   | 翰翰有39張色紙，小誠有77張色紙，兩個人一共有幾張色紙?                                |
| B   | 翰翰有39張色紙，小誠有77張色紙，兩個人 <b>一共有</b> 幾張色紙?                       |
| 原句：冰箱裡有426毫升的紅茶，媽媽又煮了2公升720毫升的紅茶，現在家裡有幾公升的紅茶? |  |
| R   | 冰箱裡有426毫升的紅茶，媽媽 <b>又</b> 煮了2公升720毫升的紅茶，現在 <b>家裡</b> 有幾公升的紅茶? |
| LS  | 冰箱裡有426毫升的紅茶，媽媽 <b>又</b> 煮了2公升720毫升的紅茶，現在 <b>家裡</b> 有幾公升的紅茶? |
| T   | 冰箱 <b>裡</b> 有426毫升的紅茶，媽媽又煮了2公升720毫升的紅茶，現在 <b>家裡</b> 有幾公升的紅茶? |
| B   | 冰箱裡有426毫升的紅茶，媽媽 <b>又</b> 煮了2公升720毫升的紅茶，現在 <b>家裡</b> 有幾公升的紅茶? |
| 原句：有一堆鳳梨，每14顆裝成一箱，共可裝成28箱，這一堆鳳梨有幾顆?           |  |
| R   | 有一堆鳳梨，每14顆裝成一箱， <b>共</b> 可裝成28箱， <b>這</b> 一堆鳳梨有幾顆?           |
| LS  | 有一堆鳳梨，每14顆裝成一箱， <b>共</b> 可裝成28箱， <b>這</b> 一堆鳳梨有幾顆?           |
| T   | 有一堆鳳梨，每14顆裝成一箱， <b>共可裝成</b> 28箱，這一堆鳳梨有幾顆?                    |
| B   | 有一堆鳳梨，每14顆裝成一箱，共可裝成28箱，這一堆鳳梨有幾顆?                             |

Table 6.2: 各簡化法刪減字比較，LS代表Label Sequence，R是Rouge-N，T是Transformer based，B是BERT based，紅字代表被刪除的字



表6.2.1是各個方法刪減字數的幾個例子，各個方法基本差異不大，但是從例子二的句子中可以發現BERT based的確傾向於刪除那些冗字，例如刪除”又”、“裡”、“共”、“的”....之類的無關緊要的介系詞，而不會過於做出太大膽的決定，像是名詞動詞此類對於句子相當關鍵的詞語。也因此在例子一跟三中，由於句子本身已經很精簡了，所以BERT based並未對句子進行更動。LS由於是取關鍵字，所以也有達到去除冗言贅字的效果。Tranformer based雖然在生成訓練資料時是以LCS計算出的刪字分佈為依據，但因為最後的機率閾值的設定，形成了結果上的差異，例如在例子三中就把關鍵字”裝成”給誤刪了。Rouge-N based以正確句子為比較對象，所以結果上較符合原來的句子，但有時候也因為句子未曾在正確題目中出現進而造成錯誤，像是表中的例子二只刪除了”家”，而非連”裡”也一併簡化。

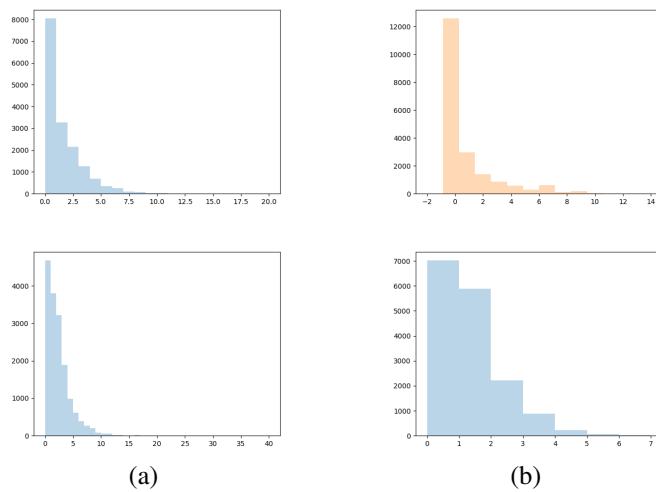


圖 6.2: 各個方法刪除字數的分佈，左上是Rouge-N，右上是BERT based，左下是Label Sequence，右下是Transformer based

圖6.2是各個方法刪減字數的分布圖，可以發現0是佔比最多的，原因是因為透過人工觀察，目前小學數學解題系統不能解決的題目多半是少數子句無法處理所導致，意味著通常一個題目裡只有少數句子是需要修改的，所以不用修改是模型遇到多數句子的最佳選擇，因此0才會是最突出的。BERT based刪減最多的字數是四種裡面最大的，原因在於距離閾值的定義，以及以平均值當作句子向量所帶來的誤差，這兩者導致距離閾值在某些情況下刪了過多的字。Label Sequence的做法分佈與其他三者最不相同，原因在於LS的做法本質上是去取key word，但是有些時候我們為了通順度的考量會加入一些冗言贅字，例如”是”或是”了”，但這些字的佔比通常不會佔據句子中太大部分，因此LS的刪字字數分佈多半集中在小



於5這個區段。

Transformer based分佈則與機率閥值的取值相關，隨著機率閥值越取越小分佈也會趨近於其他三者，如圖6.3就是在機率閥值為0.6的情況，分佈上就與其他三者類似，可以看見在0的地方特別突出，其餘的則驟降，但是又因為在訓練時有特別填positive data的權重，因此0與其他字數的分布沒那麼懸殊。

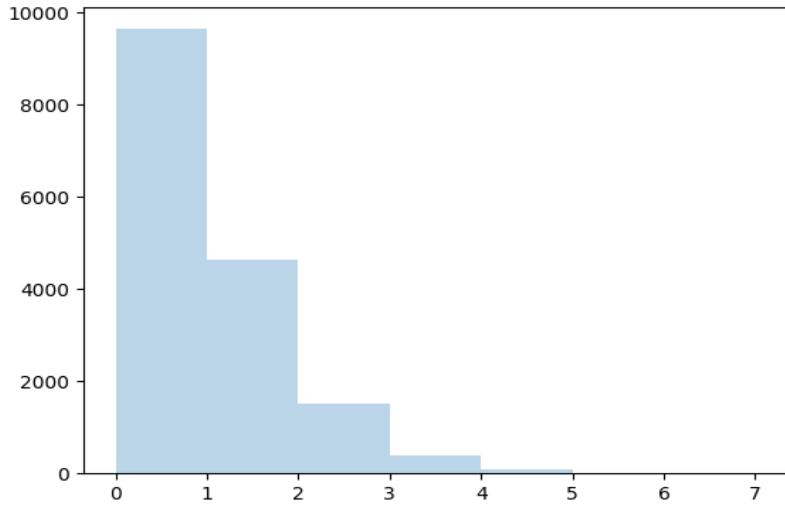


圖 6.3: Transformer based在機率閥值設為0.6時的情況

### 6.2.2 句子通順度比較

為了衡量BERT based的作法是否有依照”冗字”的原則去刪字，以及了解其餘幾種框架基底的做法刪減後留下的句子是否通順，於是按照論文[23]提到衡量句子通順度的方式來計算各個方法刪減後的句子，具體公式如6.1。

$$score = (p_{w_1} * p_{w_2} * \dots * p_{w_n})^{(\frac{-1}{n})} = (\prod_{i=1}^n P(w_i | w_1 \dots w_{i-1} w_{i+1} \dots w_n))^{\frac{-1}{n}} \quad (6.1)$$

$(w_1, w_2, \dots, w_n)$ 代表的是一個句子中的字詞，若一個句子是通順的，則遮罩其中的某個字時應該也有高機率能預測出被遮罩的字，於是一個句子的通順度就定義為依次遮罩句子中的每個字，在這樣的情況下根據上下文讓模型去預測被遮罩的字，取預測出的分布中的被遮罩的字的機率相乘。

由於是依照上下文去預測回被遮罩的字，故模型的選擇上應以Non-autoregressive的語言模型較為合適，所謂autoregressive的模型就是指那些依照



前面的字去預測下一個的模型，例如前面章節2提到的RNN以及LSTM，而non-autoregressive就是依照上下文來預測被遮罩的字的模型，例如BERT。所以我選擇以Hugging Face的transformers開源包中的BERT Masked LM模型當作衡量通順度的基底模型，結果如表6.2.2，分數越小代表產生出來的句子越通順。

在計算通順度分數時，還必須去注意小學數學題目中有很多地方是包含數字，數字的值是很難通過上下文確切的預測出來，也因此必須在計算時遮罩掉是數字的位置的機率值。

|                   | score       |
|-------------------|-------------|
| Original          | 16783.5678  |
| Rouge-N           | 19690.64453 |
| Transformer based | 20370.03320 |
| BERT based        | 18494.1406  |

Table 6.3: 各個簡化方法通順度分數表

Original代表未被改寫的原始題目。不意外的以BERT based得到最低的分數，這是因為BERT在刪字的原則上僅僅是去刪除冗言贅字，而這樣贅字在對於預測被遮罩的字詞的幫助是相當有限的，換句話說刪除這些字對於通順度的影響也很小。

Transformer based的通順度與Rouge-based方法較為接近，因為兩種方法本質上都是去找最匹配的詞性序列，又因為Transformer based的訓練資料是由正確的句子依照章節5.1.2提到的機率分佈而來，也因此兩者對於句子的方式某種程度上是相似的，也因此通順度差距不大。

這裡不去衡量Label Sequence的通順度，原因在於Label Sequence原則上是以只取句子中的關鍵字，在本質上與通順度背道而馳。

### 6.2.3 通用簡化法

前面提到對於簡化法的運用都只侷限在當作”補丁”來使用，也就是當題目跑過一次小學數學解題系統後確定答案是錯誤的情況下，我們才會在跑5提到的簡化法去做句子的精簡，期望改寫完的題目可以被小學數學解題系統所處理。然而我們是否能將這樣句子精簡的模型先套用到所有的題目，再去為小學數學解題系統所處理，讓這個模型當作是一個通用的簡化法則，讓小學數學解題系統中的簡化法則只需要針對系統所特別需要的框架設計，盡量減少窮舉的情形發生。於是



為了達到這個目的，我測試了BERT based和Label Sequence based在先套用在所有的題目上，然後再去跑小學數學解題系統的情況下依然可以答對的題數。不去比較Rouge-N based的原因在於，Rouge-N based算法上是去比對正確框架的句子。

剩餘三個作法的比較結果如表6.2.3，Transformer的做法效果遠低於其餘兩種，會造成這樣的原因推測可能有三點，1)BERT based的模型在刪除字詞時是以刪除最不重要的字為主，而這些字本身很可能已經包含在小學數學解題系統內建的簡化規則裡，因此對於整個系統影響較小。而Label Sequence是直接只取系統需要的字詞，本質上就有融合小學數學系統中的reduction機制了。2)Transformer based在做訓練時儘管是以正確題目的句子插入字數而成，但是有可能遇到這種情況：不同字卻有相同詞性，詞性排列相同的句子卻對到系統中不同的動詞框架底下，並且此詞性的字在其中一種框架必須刪除，另一種卻必須留著的情況，這樣的狀況可能源自於在編寫小學數學解題系統的框架時，沒有一個通用的法則，同一個人在編寫的時候就會有差異了，更遑論由很多語意人員一同編寫的小學數學解題系統。3)模型在訓練時鮮少能達到百分之百的準確度，這三種原因造成Transformer based在當作通用簡化法時準確度比其餘兩者低。

此外，BERT based的表現又略遜於Label Sequence，可追究於BERT based在設計上並不是完全契合我們系統中固有的reduction，BERT based作法僅自定義簡單的兩個限制，也因此比起脫胎於小學數學解題系統中的reduction的Label Sequence而言，對原本算對的題目所產生的影響也更大一些。

|      | Original | Label Sequence | BERT based | Transformer based |
|------|----------|----------------|------------|-------------------|
| 答對題數 | 1096     | 1063           | 1027       | 849               |

#### 6.2.4 Label Sequence加上小學數學解題系統分析

在上一章節中的最後，我們提到一個可以讓目前的Label Sequence與小學數學系統做結合的流程圖，並提出一個以樹的遍歷依序填值進入舊有的instance map中，所以儘管以Label Sequence生成的instance map有多出一些額外的節點也沒有關係，因為這些多出的節點在填值時會因為找不到在原本系統中的instance map的相應位置，而不會填入，所以不會影響結果。

多數情況下只要Label Sequence設計得當，融合的作法就能成功，然而當遇到需要進行改寫的句子時則會產生問題，以圖6.4為例，句子”可以裝成幾袋”在原本的系統中會對到”分配”框架，而”分配”框架底下的instance map如下左圖，右圖



是新的以Label Sequence定義的instance map，可以發現在Label Sequence的instance map中是找不到”agent“的對應點，這是因為Label Sequence中並沒有改寫的機制，也因此當遇到這種情況時兩個instance map就難以進行融合。

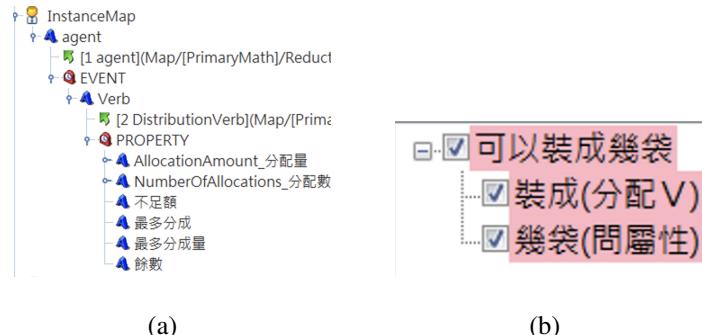


圖 6.4: 左邊是舊的”分配”框架下的instance map，右邊是新設計的instance map



# Chapter 7

## 結論與未來展望

本章節總結一下目前的研究，以及提出一些未來可以改進的方向

### 7.1 結論

在本論文中，我們提出了一個對於數學文字題(Math Word Problem)新的解決方案，並將這樣的方法付諸實行，完成一個完整的系統。然後針對此系統不能解決的題目，嘗試提出幾個方法來完善整個流程，最後導入一個新的概念，並將其與現有的系統做結合，且以數個例子說明其可行性。

我們期望透過這樣的研究能啟發後人解決問題的創新思維，讓處理數學文字題的方法不再局限於現今流行的類神經網路。

### 7.2 未來展望

底下我針對我所參與到的三個部分：小學數學系統的解釋生成、句子過濾、Label Sequence與小學數學解題系統的融合，分別提出一些可以改進的地方。

首先對於小學數學解題系統的解釋生成方面，都是人自己去判斷解釋的對錯，若能想出一個能自動衡量產生解釋的好壞的演算法，便能對解釋生成錯誤的情況進行修改，對於整個系統也有所幫助。

句子過濾的部分，因為沒有訓練資料導致在方法的設計上受到很大的限制，對於未來的改進上或許能人工標註一些資料，有了標註的資料就能進行端到端的訓練，例如在BERT based中的距離閥值以及Transformer based的機率值一直是我不斷嘗試後的結果，並且每個句子都採用同樣的值，若能透過訓練讓模型自動調



整，結果上就能更進一步。

對於BERT based作法在論文中是以BERT的character embedding的平均值為句子向量，由於BERT在訓練時並沒有特別針對句子向量去做訓練，所以若是能以其他以句子相關性去做訓練的模型(例如UKPLab的sentence-transformer)來取代論文中的BERT模型效果應該也能有所提升。

最後是Label Sequence與傳統小學數學解題系統的融合，在論文中已經提出一個初步的流程圖，並且以13種中其中幾種框架來佐證此方法的可行性，後續期望能融入更多的框架，與設計更多的Label Sequence，但是要注意的是當融入更多框架時，論文中提出的樹的融合法是否可以全部適用，這點也必須去實驗，並且如何將改寫機制引入Label Sequence中也是一個重要的問題。



# Bibliography

- [1] Sutskever and I.Vinyals and Le, Q., “Sequence to sequence learning with neural networks,” *In Advances in Neural Information Processing Systems,NIPS 2014*, 2014. ix, 5
- [2] A. Vaswani an d N. Shazeer and N. Parmar and J. Uszkoreit, L. Jones and A. N. Gomez. Kaiser and I. Polosukhin, “Attention is all you need,” *in Advances in neural information processing systems*, pp. 5998–6008, 2017. ix, 6, 7, 27
- [3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019. ix, 9, 10, 27
- [4] J. Elman, “Finding structure in time,” *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990. 4
- [5] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 5
- [6] J.Gehring, M.Auli, D.Grangier, D.Yarats, and Y.N.Dauphin, “Convolutional sequence to sequence learning,” *in Proceedings of the 34th International Conference on Machine Learning-Volume*, vol. 70, pp. 1243–1252, 2017. 6
- [7] T. Mikolov, K.Chen, G.Corrado, and J.Dean, “Efficient estimation of word representations in vector space,” *CCoRR*, vol. abs/1301.3781, 2013. 9
- [8] Daniel G Bobrow., “Natural language input for a computer problem solving system .” *In Minsky, M., ed., Semantic information processing*, pp. 146–226, 1964. 11



- [9] M. J. Hosseini, H. Hajishirzi, O. Etzioni, and N. Kushman, “Learning to solve arithmetic word problems with verb categorization,” *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 523–533, 2014. 11
- [10] N. Kushman, L. Zettlemoyer, R. Barzilay, and Y. Artzi, “Learning to automatically solve algebra word problems.” *In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics,ACL 2014*, pp. 271–281, 2014. 11
- [11] Lei Wang and Dongxiang Zhang and Lianli Gao and Jingkuan Song, Long Guo and Heng Tao Shen., “ MathDQN: Solving arithmetic word problems via deep Solving general reinforcement learning.” *In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence.*, 2018. 12
- [12] Zhipeng Xie and Shichao Sun, “A Goal-Driven Tree-Structured Neural Model for Math Word Problems,” *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence,IJCAI2019*, pp. 5299–5305, 2019. 12
- [13] Ting-Rui Chiang and Yun-Nung Chen, “Semantically-Aligned Equation Generation for Solving and Reasoning Math Word Problems.” *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, p. 2656–2668, 2019. 12
- [14] Chien-Tsung Huang and Yi-Chung Lin and Chao-Chun Liang and Kuang-Yi Hsu and Shen-Yun Miao and Wei-Yun Ma and Lun-Wen Ku and Churn-Jung Liau and Keh-Yih Su, “Designing a Tag-Based Statistical Math Word Problem Solver with Reasoning and Explanation,” *Proceedings of the 27th Conference on Computational Linguistics and Speech Processing (ROCLING 2015)*, pp. 58–63, 2015. 12, 13
- [15] Chien-Tsung Huang and Yi-Chung Lin and Keh-Yih Su, “Explanation Generation for a Math Word Problem Solver,” *Proceedings of the 27th Conference on Computational Linguistics and Speech Processing (ROCLING 2015)*, pp. 64–70, 2015. 12, 13
- [16] Peng-Hsuan Li and Tsu-Jui Fu and Wei-Yun Ma, “Why Attention? Analyze BiLSTM Deficiency and Its Remedies in the Case of NER,” *AAAI2020*, 2019. 25



- [17] Ming Zhong and Pengfei Liu and Danqing Wang and Xipeng Qiu and Xuanjing Huang, “Searching for Effective Neural Extractive Summarization: What Works and What’s Next,” *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, p. 1049–1058, 2019. 27
- [18] Shi Feng and Eric Wallace and Alvin Grissom II and Mohit Iyyer and Pedro Rodriguez and Jordan Boyd-Graber, “Pathologies of Neural Models Make Interpretations Difficult,” *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, emnlp 2018*, pp. 3719–3728, 2018. 28
- [19] Javid Ebrahimi and Anyi Rao and Danial Lowd and Deijing Dou, “HotFlip:White-box adversarial examples for text classification,” *In Proceedings of the Association for Computational Linguistics*, 2017. 28
- [20] Jiwei Li and Will Monroe and Daniel Jurafsky, “Understanding neural networks through representation erasure,” *arXiv preprint arXiv 1612.08220*. 28
- [21] Leila Arras and Franziska Horn and Gregoire Montavon and Klaus-Robert Muller and Wojciech Samek, “Explaining predictions of non-linear classifiers in NLP,” *In Workshop on Representation Learning for NLP*, 2016. 28
- [22] Yiming Cui and Wanxiang Che and Ting Liu and Bing Qin and Ziqing Yang and Shijin Wang and Guoping Hu, “Pre-Training with Whole Word Masking for Chinese BERT,” *arXiv preprint arXiv:1906.08101*, 2019. 29
- [23] Cunxiang Wang and Shuailong Liang and Yue Zhang and Xiaonan Li and Tian Gao, “Does it Make Sense?And Why?A Pilot Study for Sense Making and Explanation,” *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, pp. 4020–4026, 2019. 38