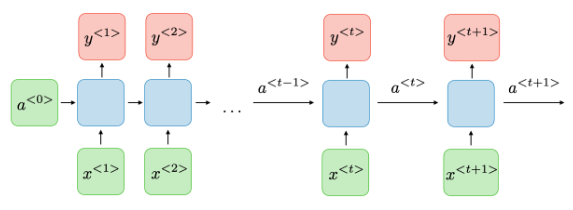
1. COCO data
2. Text

   Description automatically generatedText

   Description automatically generated我們在這裡載入Microsoft COCO dataset，並且記錄一些特殊token的index
3. A person in a suit

   Description automatically generated with low confidenceText

   Description automatically generated所有每個data都是一個image和標題用來描述image
4. RNN

* RNN的結構如下圖，每個input x都有一個上標t，t用來代表時間、第幾個token等等，期之間有上下文關係，因此每個input透過forward除了會產生結果y還會產生一個a傳給下一個節點進行計算，透過這種方式每個輸出都會與其輸入及前面的輸入及輸出相關。

1. rnn\_step\_forwad
   * Text

     Description automatically generated這邊是在implement前面藍色框框所產生a的部分，在這邊稱為next\_h，其計算方式，最後再加上一個tanh activation function。
2. rnn\_step\_backward
   * 這邊我參考了 <https://towardsdatascience.com/backpropagation-in-rnn-explained-bdf853b4e1c2#:~:text=You%20see%2C%20a%20RNN%20essentially,where%20they%20are%20summed%20up.> 所繪製的computation graph
   * Dtanh是計算tanh對input x的微分，其結果是1 - tanh^2(x)，最後在乘上backprop的dnext\_h
   * 其他的部分如computaional graph中所表示。

Text

Description automatically generated

1. rnn\_forward
   * Text

     Description automatically generated這邊與前面rnn\_step\_forward的差別在於step\_forwar只針對一連串input的其中一個做process，rnn\_forward則是對所有input都做forward的動作。透過迴圈的方式從第0個開始執行，直到T-1個(總共有T個input)，每個input的output h都會成為下一個step forward的input h。
2. rnn\_backward
   * Text

     Description automatically generated這邊也是透過前面的rnn\_step\_backward來計算。但由於是backprop所以會透過反向的迴圈計算。這邊要特別注意rnn\_step\_backward的第一個參數不能只使用dh，因為dh只是從該word產生loss propagate回來，我們還需要加上後面word產生loss所propagate回來的部分。dx的shape是(N, T, D)，所以每迴圈一次都會產生對應t的dx。在每個t上都是用相同的Wx, Wb, b，因此dWx, dWb, db的計算會是把迴圈中每輪產生的相加。dh0則是使用由後往前最後產生的\_dprev\_h。

1. RNN class
   * 在class的\_\_init\_\_中初始化所需的參數。
   * forward及step\_forward都是使用前面所說明的函數。

Text

Description automatically generated

1. WordEmbedding Class
2. Text

   Description automatically generated這個主要再把一連串文字的每個word根據W\_embed轉換成word vectors代表其features
3. FeatureExtractor
4. 透過pretrain的mobilenet\_v2來擷取每張image的features

Text

Description automatically generated

1. Captioning RNN
2. 此Class在訓練一個能夠根據照片產生其對應的描述
3. \_\_init\_\_用來初始化所需要使用的各class及變數
   * word\_to\_idx及idx\_to\_word都用來將word與index進行轉換 。
   * vocab則是所有word的資料庫。
   * null, -start, \_end, ignore\_index用來記錄一些特定的word，以便做特殊處理。
   * Text

     Description automatically generatedfeature\_extractor為一個instance用來擷取image的特徵 。 feature\_transform\_fc為一個Linear model用來將image的特徵轉換成RNN可以接收的形式。word\_embedding用來將word轉換成對應的word vectors(word的features)。Rnn則為前面所介紹的model。Output\_transform\_fc為一個Linear model用來將RNN的hidden state轉換成vocab的機率。
4. forward
   * 首先透過feature\_extractort產生image的feature。接下來，透過feature\_transform\_fc將feature轉換成RNN可以接收的hidden state。
   * Text

     Description automatically generated在RNN的部分，除了會接收上述轉換的hidden state，同時還會透過word\_embedding將將標籤所給予caption中的每個word轉換成word vector。RNN會輸出每個word所產出的hidden state，透過output\_transform\_fc將這些hidden state轉成vocab的機率，最後透過temporal\_softmax\_loss ( cross\_entropy )來計算其Loss。
5. training
   * 在training的部分，optimizer使用torch提供的Adam
   * 這邊並沒有使用我們所implement的backward，而是使用pytorch內建的autograd

Text

Description automatically generated

1. overfit small data
   * Text

     Description automatically generated在這邊使用batch\_size = 5 及 learning\_rate = 8e-4可以讓Loss降低到0.1。

Chart, histogram

Description automatically generated

1. sample
   * 此method是在test的時候用來產生圖片對應的caption。
   * 首先，會從image抓取feature並轉換成RNN的input state。由於words的開頭都是 <start>，所以一開始先建立一個(N, 1)的tensor，其index都是指向<start>。
   * Text

     Description automatically generated在迴圈的部分，會先將目前的word透過word\_embedding轉換成word vector。將feature所產生的input state及word vector透過RNN 的step\_forward產生下個hidden state。此hidden state會透過output\_transform\_fc轉換成每個vocab的分數，分數最大的就是所產生的word。
2. Train the net and show result

Graphical user interface

Description automatically generated with medium confidence

A person surfing on a wave

Description automatically generated with medium confidenceA picture containing text, snack food, sandwich, different

Description automatically generated