CNN-LSTM Architecture and Image Captioning

Shweta Pardeshi · Follow
Published in Analytics Vidhya · 5 min read · Nov 23, 2019

This post is co-authored by Kanishkkalra.

Deep learning is one of the most rapidly advancing and researched field of study that is making its way into all of our daily lives. It is simply the application of artificial neural networks using heavy and high-end modern hardware. It allows the development, training, and use of neural networks that are much larger (more layers) than was previously thought possible. There are thousands of types of specific neural networks proposed by researchers as modifications or tweaks to existing models. Some of the more prominent ones as CNN's and RNN's.

Convolutional Neural Networks were designed to map image data to an output variable. They have proven so effective that they are the go-to method for any type of prediction problem involving image data as an input.

Recurrent Neural Networks, or RNNs, were designed to work with sequence prediction problems. Some of these sequence prediction problems include one-to-many, many-to-one, and many-to-many.

LSTM networks are perhaps the most successful RNN's as they allow us to encapsulate a wider sequence of words or sentences for prediction.

The CNN-LSTM Model

One of the most interesting and practically useful neural models come from the mixing of the different types of networks together into hybrid models.

EXAMPLE

Consider the task of **generating captions for images**. In this case, we have an input image and an output sequence that is the caption for the input image.

Can we model this as a one-to-many sequence prediction task?

Yes, but how would the LSTM or any other sequence prediction model understand the input image. We cannot directly input the RGB image tensor as they are ill-equipped to work with such inputs. Input with spatial structure, like images, cannot be modeled easily with the standard Vanilla LSTM.

Can we extract some features from the input image?

Yes, this is precisely what we need to do in order to use the LSTM architecture for our purpose. We can use the deep CNN architecture to

extract features from the image which are then fed into the LSTM architecture to output the caption.

This is called the CNN LSTM model, specifically designed for sequence prediction problems with spatial inputs, like images or videos. This architecture involves using Convolutional Neural Network (CNN) layers for feature extraction on input data combined with LSTMs to perform sequence prediction on the feature vectors. In short, CNN LSTMs are a class of models that are both spatially and temporally deep and sit at the boundary of Computer Vision and Natural Language Processing. These models have enormous potential and are being increasingly used for many sophisticated tasks such as text classification, video conversion, and so on. Here is a generic architecture of a CNN LSTM Model.

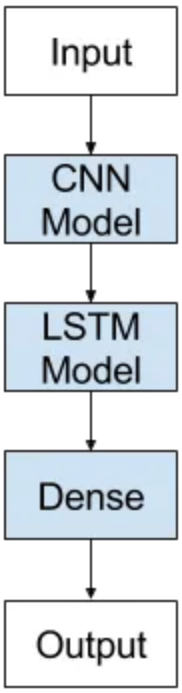

Image Source: https://machinelearningmastery.com/cnn-long-short-term-memory-networks/

Image Captioning

Describing an image is the problem of generating a human-readable textual description of an image, such as a photograph of an object or scene. It combines both computer vision and natural language processing together.

Neural network models for captioning involve two main elements:

1. Feature Extraction.
2. Language Model.

The rest of the article will elucidate our thoughts and observations while implementing a CNN-LSTM model for image captioning. Note that this post is not a tutorial on image captioning implementation but is aimed at exploring the CNN-LSTM architecture and its practical usage. The code was written in python3 and implemented in Keras. Here are the necessary requirements and pre-requisite for you to be able to understand the implementation completely. If you are interested in the implementation tutorial, you can go to https://bit.ly/2XFCEmN.

```
from os import listdir
from pickle import dump
from keras.applications.vgg16 import VGG16
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import preprocess_input
from keras.models import Model
from numpy import array
from pickle import load
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.utils import to_categorical
from keras.utils import plot_model
from keras.models import Model
from keras.layers import Input
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Embedding
from keras.layers import Dropout
from keras.layers.merge import add
from keras.callbacks import ModelCheckpoint
from numpy import argmax
from pickle import load
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import load_model
from nltk.translate.bleu_score import corpus_bleu
```

gistfile1.txt hosted with ❤ by GitHub view raw

The dataset used can be downloaded using the following links.

Flickr8k_Dataset(Contains 8092 photographs in JPEG format) — https://bit.ly/35shVWb

Flickr8k_text(Contains a number of files containing different sources of descriptions for the photographs.) — https://bit.ly/2DcBAgF

The dataset has a pre-defined training dataset (6,000 images), a development dataset (1,000 images), and test dataset (1,000 images).

The dataset information as well as data preparation for the model can be seen in the same link above.

Here, we will only show the important snippets of the code that has been used to create and run the model. You are encouraged to use a different dataset and prepare your dataset accordingly.

Feature extraction

The feature extraction model is a neural network that given an image is able to extract the salient features, often in the form of a fixed-length vector. A deep convolutional neural network, or CNN, is used as the feature extraction submodel. This network can be trained directly on the images in your dataset. Alternatively, you can use a pre-trained convolutional model as shown.

```
# extract features from each photo in the directory
def extract_features(directory):
        model = VGG16()
        model.layers.pop()
        model = Model(inputs=model.inputs, outputs=model.layers[-1].output)
        features = dict()

        for name in listdir(directory):
                filename = directory + '/' + name
                image = load_img(filename, target_size=(224, 224))
                image = img_to_array(image)
                image = image.reshape((1, image.shape[0], image.shape[1], image.shape[2]
                image = preprocess_input(image)
                feature = model.predict(image, verbose=0)

                image_id = name.split('.')[0]
                features[image_id] = feature
        return features
```

Feature Extraction hosted with ❤️ by GitHub view raw

Language Model

For image captioning, we are creating an LSTM based model that is used to predict the sequences of words, called the caption, from the feature vectors obtained from the VGG network.

```
def model(vocab_size, max_length):

        input1 = Input(shape=(4096,))
        drop_1 = Dropout(0.5)(inputs1)
        dense_1 = Dense(256, activation='relu')(fe1)

        inputs2 = Input(shape=(max_length,))
        embed = Embedding(vocab_size, 256, mask_zero=True)(inputs2)
        drop_2 = Dropout(0.5)(se1)
        dense_2 = LSTM(256)(se2)

        decoder1 = add([fe2, se3])
        decoder2 = Dense(256, activation='relu')(decoder1)
        outputs = Dense(vocab_size, activation='softmax')(decoder2)

        model = Model(inputs=[inputs1, inputs2], outputs=outputs)
        model.compile(loss='categorical_crossentropy', optimizer='adam')
        model.summary()
        return model
```

gistfile1.txt hosted with ❤️ by GitHub view raw

The language model is trained for 20 epochs. You can play around with other parameters and tune them as much as you want. We are displaying here one result that we obtained after training our network.

Generated Caption: Two girls are playing in the water.

Conclusion

A CNN-LSTM architecture has wide-ranging applications as it stands at the helm of Computer Vision and Natural Language Processing. It allows us to use state of the art neural models for NLP tasks such as the transformer for sequential image and video data. At the same time, extremely powerful CNN networks can be used for sequential data such as the natural language. Hence, it allows us to leverage the useful aspects of powerful models in tasks they have never been used for before. This post was just to introduce the concept of hybrid neural models and encourage the people to increasingly use different architectures of the CNN-LSTM models.

References

VGG feature extraction by pre-trained model

Download Open Datasets on 1000s of Projects + Share Projects on One Platform. Explore Popular Topics Like Government...

www.kaggle.com

NLP-2019

Basic Probability & Statistics (ES 331/ MA 202) or equivalent Basic understanding of Python programming (ES 102/ ES...

sites.google.com

How to Automatically Generate Textual Descriptions for Photographs with Deep Learning

Captioning an image involves generating a human readable textual description given an image, such as a photograph. It...

machinelearningmastery.com

When to Use MLP, CNN, and RNN Neural Networks

What neural network is appropriate for your predictive modeling problem? It can be difficult for a beginner to the...

machinelearningmastery.com

How to Develop a Deep Learning Photo Caption Generator from Scratch

Develop a Deep Learning Model to Automatically Describe Photographs in Python with Keras, Step-by-Step. Caption...

machinelearningmastery.com

CNN Long Short-Term Memory Networks

Gentle introduction to CNN LSTM recurrent neural networks with example Python code. Input with spatial structure, like...

machinelearningmastery.com

Machine Learning NLP Computer Vision

Written by Shweta Pardeshi

Follow

43 Followers · Writer for Analytics Vidhya

ML Engineer at Granular.ai | Educative Author | 30k+ views on Medium | Analytics Vidhya Author | IIT Gandhinagar | https://www.buymeacoffee.com/shwetapar1

More from Shweta Pardeshi and Analytics Vidhya