Journal of Ubiquitous Computing and Communication Technologies (ISSN: 2582-337X)
www.irojournals.com/jucct/

Image Captioning in Tamil Language using Encoder-Decoder Architecture

Thivaharan. S[1], Srivatsun.G[2], Pranav Kiran. S[3], Johan Benoni Raul. J[4]

[1]Asst. Prof (SG), Dept. of CSE, PSG Institute of Technology and Applied Research, Coimbatore-641062

[2]Associate Professor, Dept. of ECE, PSG College of Technology, Coimbatore - 641002

[3, 4]UG scholar, Dept. of CSE, PSG Institute of Technology and Applied Research, Coimbatore-641062

E-mail: [1]thivaharan.s@gmail.com,[2]gsn.ece@psgtech.ac.in,[3]d20z106@psgitech.ac.in,[4]d20z215@psgitech.ac.in

Abstract

Image captioning is the process of using clear, meaningful words to describe the characteristics of an image. This feature has wide applications in social networking applications such as Facebook and Instagram, and video streaming platforms such as YouTube and Netflix, where the need to verbalize an image or video is evident. Image captioning is also one of the most requested features in next-generation AI systems. It has huge applications in the Deep Learning domain. Much research is actively being done on image captioning, which can solve a good deal of real time problems such as the need for a system that can aid visually disabled people, creating effective captions that can be incorporated in self-driving vehicles, etc. This elaborate yet useful feature can be incorporated with the help of various technical concepts such as Natural Language Processing, Computer vision, Image Processing, etc. The image captioning feature has already been attempted on English language and with the help of extensive research and technical advancements these attempts have been fruitful and successful. Nowadays, there are many applications and models available based on image captioning of English language. This has paved a path for further advancements in this domain. A lot of research are now being undertaken to incorporate this highly useful feature with non-English languages. English being the native language for a relatively smaller proportion of people, it would be helpful for people whose native language is not English, to get their images captioned in the language of their choice. This research focuses on image captioning in Tamil language and its underlying methodology and architecture. Moreover, the paper also includes

DOI: https://doi.org/10.36548/jucct.2023.1.003
Received: 05.01.2023, received in revised form: 05.02.2023, accepted: 20.02.2023, published: 02.03.2023

experiments related to this with the help of an image captioning model which uses a combination of Convolution Neural Network and Long Short -Term Memory models.

Keywords: Image Captioning in Tamil, Convolution Neural Network (CNN), Long Short -Term Memory (LSTM), Natural Language Processing (NLP), Computer Vision, Image Processing, Artificial Intelligence, Deep Learning

1. Introduction

The sequential process of providing a textual description for the given input image is known as Image Captioning. The description can contain sentences related to the most prominent features of the image such as the characteristics of an entity or any actions performed by the entity. It can also be sentences depicting the image as a whole feature. This is basically image processing, which is extensively used in Machine learning and Deep learning algorithms. On a more detailed note, this process uses Natural Language Processing (NLP) and Computer Vision for making this possible. Image captioning can be embedded in applications and models that can have a significant impact on various areas such as social media analytics, recommendation engines, virtual assistants, and more. There has been a lot of research on image captioning in English because of large image captioning datasets like MS-COCO and Flickr. However, little development has been done for non-English languages. This is a challenge for people whose first or spoken language is not English. This is due to the high complexity associated with non-English languages. The basic grammar of non-English languages such as Tamil is very different from English. Also, the datasets of alphabets and words needed for languages like Tamil are very large compared to the small datasets needed for English.

The goal of this research is to build a model that can generate image captions in Tamil. The whole process can be divided into sub-processes. The first is to process the image and generate a corresponding caption in English. The second is to translate the generated captions into Tamil. Difficulties associated with using Tamil can be solved using deep learning algorithms. It has already been demonstrated that in many cases more accurate and efficient predictions can be made using neural networks instead of traditional machine learning algorithms. Convolutional Neural Networks (CNNs) are the best option when it comes to image processing. Therefore, it is more appropriate and efficient to do the first task using a CNN model. As a second task, Recurrent Neural Networks (RNNs) could be a promising option. However, when it comes to accuracy, Long Short -Term Memory Models (LSTMs) are a better

choice. LSTM models not only predict outcomes when given current logistic, but also act rationally considering the past predictions. Therefore, LSTM is obviously the best choice to solve the second problem.

2. Literature Studies

Picture captioning can be an outstanding field with many promising applications, including processing verbal communication. Some remarkable advances in English, Chinese [9], Turkish (Yılmaz) and Arabic (Al-Museini H.A. and Benhidur H., 2018) have mostly been implemented using deep learning to handle the myriad of complexities [5]. Several deep learning methods have achieved good results in terms of performance and accuracy in the image captioning task, due to the considerable amount of knowledge in the various datasets. Some of these technologies have their own addresses for English [3]. In other language speaking countries, LSTM which is a special kind of RNN, one of the most widely used deep learning techniques [4]. Flickr dataset and MS-COCO English, attempt to generate image captions in several non-English languages such as Arabic, Turkish and Chinese. The paper used a superior technique to generate image caption datasets for the language [6]. Arabic versions of the Flickr and MS-COCO datasets have been created [11]. The work included a hybrid LSTM-CNN model in the dataset. Building the model required additional use of the infamous "merge" design, and the system has shown promising results.

Alternative analysis studies have shown that results can be significantly improved at the expense of large numbers of large datasets. This is often due to the fact that Arabic can be a morphologically complex language compared to English. The method described in [] includes an encoder-decoder version. It is heavily influenced by the illustration by Viñals O which consists of CNN and RNN models for the two main image captioning functions. Part of the CNN version was used for extracting parameters from the dataset, from which a RNN model was used to generate Turkish captions. To obtain the captions in Turkish, various translation models were created, mostly copy-based on the dataset. This approach is similar to the work deployed in Arabic, except for differences in design, striking similarities in language complexity, and promising performance of the encoder-decoder architecture in the case of Turkish datasets. A constant model is selected for image captioning in Tamil [10].

Zhang X described an iterative RAL model for Chinese image captions. The model used Inception-v4 to collaboratively extract features. The mechanism in the RAL version determined the individual operating weight. All modes used in the work were based on the CNN-LSTM framework for feature extraction related to signature generation, with many variations. Also known as Encoder-Decoder, this commonly used CNN-LSTM architecture was originally designed by Vinyals. It is based entirely on a CNN model that acts as a model, which is encoder, which is then followed by a model known as RNN model that generates titles in English language to become a decoder. The above-mentioned captioning, utilize translation in device level and commits with humans to construct and fine-tune the given dataset, then the deep learning model is applied to the given dataset. This is attributable with the steady performance among other languages. The given structure is additionally referred as 'Neural photograph Captioning' technology, which is employed within the model. The implications of this model are in complete distinction to the evaluations of the Merge architecture. The stipendiary of various other works for the shortage of images in various languages through efficient ways equivalent to mismatched Image Captioning technology, with the help of Language Pivoting and Image Captioning victimization trilingual records, permits to create the usage of datasets in English, which suits their necessities. Moreover, those methods usually consist of parallel corpora on the Brobdingnagian scale which isn't feasible in many cases considering the provision of sources for this venture at that stage. Gu J has attempted a different method of taking photos, which involves getting the properties of a photo captioning element by victimizing the supply-target parallel corpus from the source language to the target language.

The proposed framework consists of a hybrid model that describes pictures inside the pivot language and any other model such as Neural gizmo Translation model, etc. in order to convert sentences from source to destination language. According to various sources, it has been assumed that English is the foremost helpful resource-scarce [5] destination language, whereas Chinese language is aid-wealthy because of the source language. Upon estimation, the results have outperformed the traditional ways on datasets. Generally, the German captions had to be manually created for the image in situ of translation of the dataset in English. In the case of Tamil, the requirement of manually obtained set of datasets is suggested.

3. Proposed Methodology

3.1 Caption Generation

This module uses an encoder-decoder architecture to generate titles for images. This architecture consists of a CNN model that meets the requirements of an image encoder. For image classification, this model is first pre-trained before being activated as an encoder. The hidden layer, which is also the last layer, is used as input to the LSTM model, which in turn acts as a decoder to generate the signature.

The process of writing a description of an image is called image captioning. Labels are generated using both computer vision and natural language processing. [Image ——> Caption] becomes the dataset format [12]. The input photos and corresponding output captions make up the dataset. The CNN model described above acts as an encoder. When an input image is fed into a CNN model, the desired features are extracted from the encoder.

The last hidden layer of this CNN model connects to the top layer of the decoder, i.e., the top LSTM model. The decoder model consists of a set of RNNs that perform language modelling. This process is carried out at the word level. When this process is run for the first time, <START> vector is used.

Training

Key parameters used are:
x1 = "START"
y1 = first word
x2 = first word vector
yT = <END> Tokens
xT = last word

The image of the instance is the input, and the expected output is a reasonable image description. Therefore, this must be done one word at a time. The initial decoder time step receives the output from the most recent hidden state of the CNN (encoder). Given the desired label y1 in the sequence, the vector x1 is defined. Similarly, x2 is set and the network is expected to predict the second word. The target label yT is used in the last step, where xT = "last word" [1].

Testing

The decoder start time step is the representation of the image passed to the decoder. The distribution is computed over the first word y1 using the vector x1. To construct yT, the word patterns available in the distribution are used (otherwise, argmax is chosen) and the embedding vector is changed to x2. Then, the process is repeated.

During testing, the decoder output at time "t" is returned to the decoder and used as input for time "t+1". Therefore, the procedure takes an input that is an image and generates the first word, then uses the input image and the first word to generate a second word, then uses the input image and the first two words to generate a third word, and so on. Therefore, the n^{th} word is generated from the input image and the previous n-1 words. Now, another level of complexity is added to the problem. In addition to the previous word, a condition has been added that the next word must be generated using the input image.

As before, the probability of predicting the 't^{th}' word is given by:

$$P(y_t|y_1^{t-1}) \tag{1}$$

Given a word, an input image and the previous word, the probability is given by:

$$P(y_t|y_1^{t-1}, I) \tag{2}$$

Some terms in y1, y2, ..., y(t-1) are removed when the input is encoded into a single representation before computing the final output distribution. This is the RNN state vector "st" during the time step "t". This is because it depends on inputs from the previous time steps. This methodology should also be applied to this scenario. That is, any word can be encoded into an encoded representation, and images can also be encoded as vectors. This task can be solved with a simple CNN model. Considering that the image goes through the VGG-16 model on the output layer and before the last concatenated layer, image representations can be selected from any layer, but it is always a good idea to consider output coming from deep layers. This is why the image representation is ultimately chosen. The final output of 'y' must be a function of 'st', a shorthand representation of all previously generated words and a representation of the input image, i.e., both the vector representations.

"y2" is now calculated as:

y2 = O (V*s2 + c)

This means that y2 depends on s2, which in turn depends on s1, x1 and s0 [6]. However, s0 is nothing more than the encoded form of the image. Therefore, it can be seen that y2 depends not only on y1 but also on I, the input image.

4. Complete Architecture

A CNN encoder model encodes its input because when an image is given as input, an encoded version of the image is obtained. Acting as a decoder, the LSTM model takes this encoded version of the image, returns it as input at each step, and then decodes the encoded information one word at a time, to produce the output. The state vector acts as an input to this feedforward network and the distribution of the softmax function is its output.

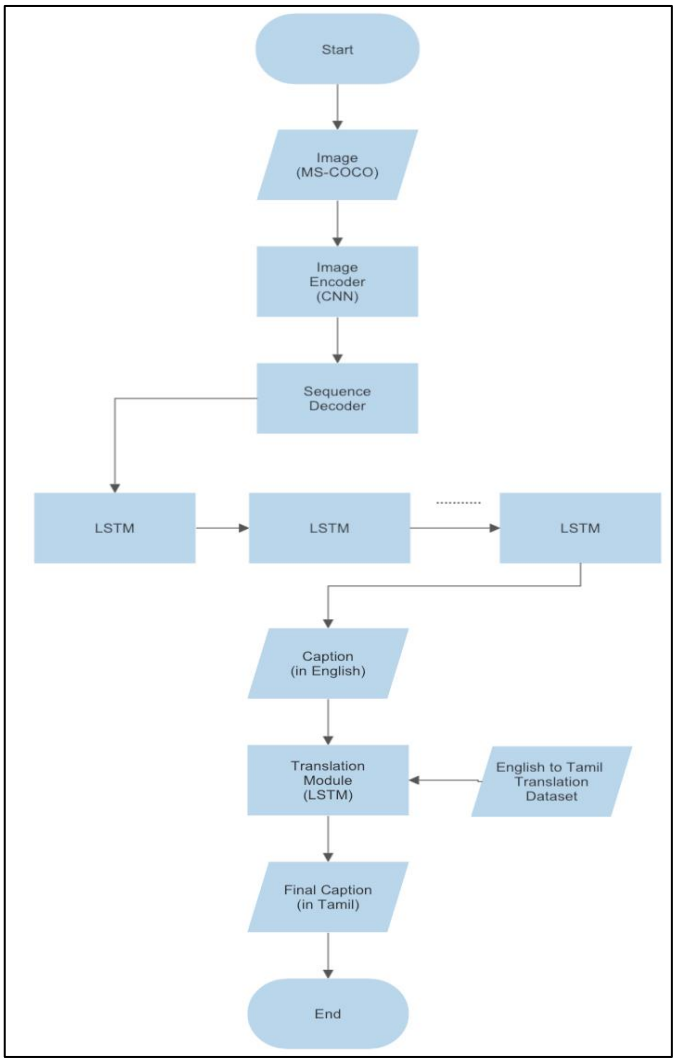


Figure 1. Flowchart of Image Captioning and Translation Model

5. Components of image captioning

Task: Creating image captions

The variable "x" represents one image as input, and the output "y" can represent multiple words. The result obtained is the serial output.

- **Task:** Image captioning
- **Data:** $\{x_i = image_i,\ y_i = caption_i\}_{i=1}^{N}$

Data: In this scenario, the data represents the number of images with correct descriptions/titles.

Model: The output of the proposed model is a function of the input, and produces an output at each time step. Therefore, it is specified as:

- **Model:**
  - **Encoder:**

$$s_0 = CNN(x_i)$$

  - **Decoder:**

$$s_t = RNN(s_{t-1}, e(\hat{y}_{t-1}))$$
$$P(y_t|y_1^{t-1}, I) = softmax(Vs_t + b) \quad (3)$$

The input "xi" is passed to the CNN model and encodes the information. It then transmits the encoded information of this image, and at each time step a single encoding information of the previous RNN output is obtained. This provides the status value at a specific time step. This is done using the final state of the distribution.

Parameters: It includes all RNNs, CNNs, offsets, etc.

- **Parameters:** $U_{dec},\ V,\ W_{dec},\ W_{conv}, b$

Loss: After receiving loss, all parameters can be updated according to algorithm update rule.

- **Loss:**

$$\mathscr{L}(\theta) = \sum_{i=1}^{T} \mathscr{L}_i(\theta) \quad = -\sum_{i=1}^{T} \log P(y_t = \ell_t | y_1^{t-1}, I) \quad (4)$$

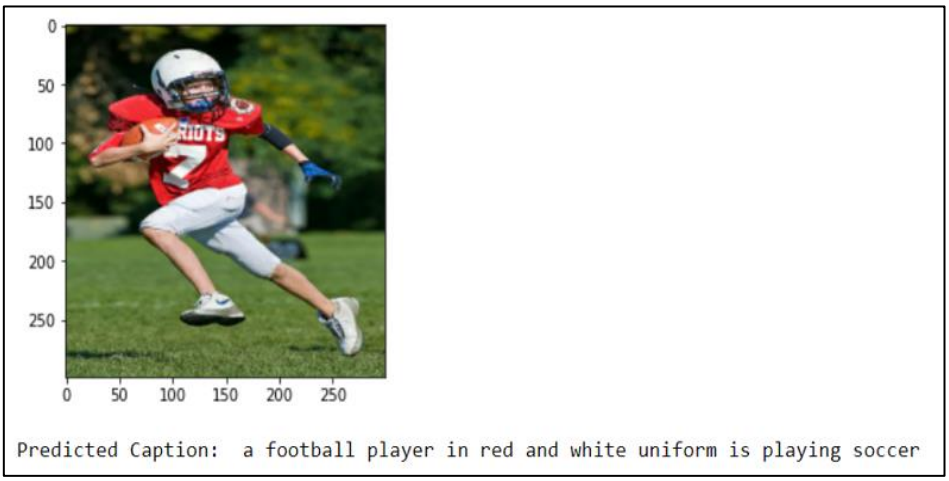


Figure 2. Output for Image Caption generation –Example 1

Figure 3. Output for Image Caption generation –Example 2

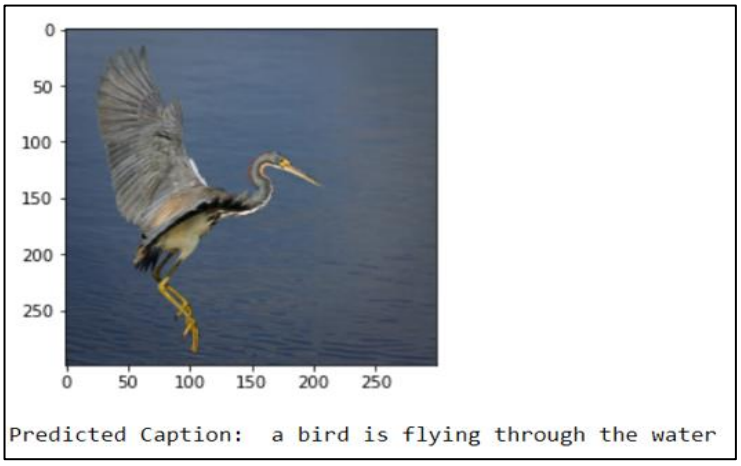


Figure 4. Output for Image Caption generation –Example 3

6. Caption Translation

Images are translated in this module using an LSTM model which is a unique RNN model that incorporates memory blocks and fundamental logic gates to address the issue of long collection dependencies that RNNs have. Also, it makes the RNN more adept at managing lengthy series of information. An additional Encoder and Decoder model group is included in the transducer model. Several encoder or decoder modules are layered on top of one another to form an encoder-decoder structure.A multi-head attention layer and a fully connected feedforward layer are present in every module. Another way to remember the location facts of the input series is required because RNNs are no longer useful. The transducer model also employs a positional embedding method to add a relative position to each component of the input dataset. These positional statistics are then utilized to characterize each sphere. The output vector of the LSTM model has a fixed dimension [6] based on the estimation made by the LSTM model above. In order to encrypt variable-length strings in the source language, the same dimension vector is used. It is similarly simple to use the conventional version of LSTM when translating into English because the input consists of variable-length English sequences. Nevertheless, the model does not entirely accept English input sequences without error. As a result, the translation effect is inadequate. Also, using rigorous and sophisticated dimensional images of the input models, i.e., the equal level of interest to the series, is not appropriate for raising the degree of interpretation due to various features of translation. Hence, an interest mechanism is added to the LSTM model in an effort to address the aforementioned issues, and an English system translation version that is entirely based on LSTM attention embedding is deployed.

Initially, the supply language collection is represented using a set of vectors rather than a strict measurement. The translation model is then advanced to pay more attention to the parts that have a high relevance to the supply language later in the interpretation system by applying dynamic selection of the preceding vectors throughout the target series technology operation. This has an impact that ultimately enhances the version's overall translation performance. The LSTM-English system translation model has three parts: an encoder, a decoder, and an attention model, in addition to the percentage mechanism. The following hidden layer is calculated in the same manner as the LSTM decoder component on the version target side, as illustrated below:

$$Z_i + 1 = \sigma\left(c_i, u_i, z_i\right), \tag{5}$$

In LSTM models with integrated attention processes, the background vectors [9] represent various sets of vectors and are not evenly stable. A different background vector can be allocated to each word in the target language sequence. Considering that encoder-implicit j's layer is in the state h_j, then the background vector corresponding to it can be calculated by:

$$C_i = \sum_{j=1}^{r} a_{ij} h_j, \tag{6}$$

where the weight is represented by a_{ij}. The below equations can be used to compute this:

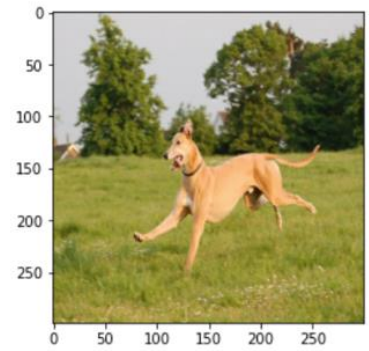
$$a_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{k=1}^{T} \exp\left(e_{ik}\right)},$$
$$e_{ij} = a\left(z_i, h_j\right), \tag{7}$$

Here, "a" represents the function that quantifies the likelihood, that the current state h_j of the destination language and the source language states will match. It can be calculated by:

$$e_{ij} = v^T \tanh\left(W_{zzi} + W_h h_j\right), \tag{8}$$

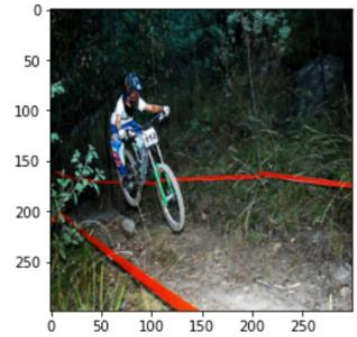
where the parameters that the model must learn are represented by v, Wz, and Wh.

Within the destination language state, the words can locate a different associated historical past vector since the background vectors of the LSTM version is integrated with the interest mechanism. The length-variety dependence issue that plagues many LSTM models can be resolved by leveraging the embedded attention mechanism included into the LSTM design, which enhances the model's overall performance by assigning the model extraordinary weights at the source language region.



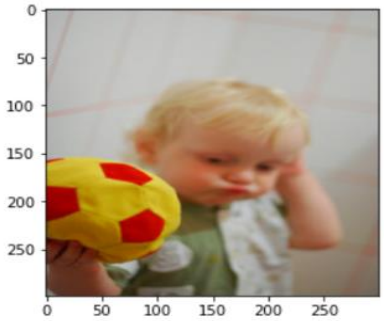
Predicted Caption: a brown dog is running through a field in a field
Translated Caption: ஒரு பழுப்பு நிற நாய் வயல்வெளியில் ஓடுகிறது

Figure 5. Output for Caption translation –example 1



Predicted Caption: a man riding a bike through a forest
Translated Caption: ஒரு மனிதன் காடு வழியாக பைக்கில் செல்கிறான்

Figure 6. Output for Caption translation –example 2



Predicted Caption: a baby is playing with a toy
Translated Caption: ஒரு குழந்தை பொம்மையுடன் விளையாடுகிறது

Figure 7. Output for Caption translation –example 3

3.1 Conclusion:

This work is mainly devoted to a preliminary study of the feasibility of using Tamil captions for images. In this study, the encoder-decoder model structure previously used for Turkish captioning, has been employed. The encoder-decoder model is a variant of the hybrid CNN-LSTM model. There are certain limitations to signature generation with respect to model misidentification of individual unique features that are not present in the training dataset. The

results of this study strongly suggest that model accuracy and performance can be improved with more defined datasets.

4.1 References:

[1] Thivaharan.S, Srivatsun.G, "Keras Model for Text Classification in Amazon Review Dataset using LSTM", Journal of Artificial Intelligence and Capsule Networks (IROAICN), June 2021, Vol.03, Issue.02, pp.72-89,ISSN: 2582-2012

[2] https://www.analyticsvidhya.com/blog/2021/12/step-by-step-guide-to-build-image-caption- Generator using deep learning/

[3] https://www.researchgate.net/publication/347970207_Image_Captioning_Using_Deep_Convolutional_Neural_Networks_CNNs

[4] http://ir.kdu.ac.lk/bitstream/handle/345/5209/11.pdf?sequence=1&isAllowed=y

[5] https://towardsdatascience.com/a-guide-to-image-captioning-e9fd5517f350

[6] Thivaharan. S."An Improved Sentiment Extraction Model for Social Media Contents using spaCy Based Deep Neural Networks", Volume 9, Issue VII, International Journal for Research in Applied Science and Engineering Technology (IJRASET) Page No: 322-327, ISSN : 2321-9653, www.ijraset.com

[7] https://medium.com/analytics-vidhya/how-to-translate-text-with-python-9d203139dcf5

[8] https://towardsdatascience.com/image-captioning-in-deep-learning-9cd23fb4d8d2

[9] https://www.researchgate.net/publication/342860841_EncoderDecoder_Architecture_for_Image_Caption_Generation

[10] https://ieeexplore.ieee.org/document/9137802

[11] https://prvnk10.medium.com/encoder-decoder-model-for-image-captioning-e01c9392ea7f

[12] https://www.academia.edu/32840609/English_To_Tamil_Machine_Translation_System_Using_Parallel_Corpus