# Image Caption generation using Encoder- Decoder and Transformers

Aman Goyal GitHub Link Venkata Krishna Kapardhi Dendukuri

## Abstract

*In recent years, neural networks have fueled dramatic advances in image captioning. Researchers are looking for more challenging applications for computer vision and Sequence to Sequence modeling systems. They seek to describe the world in human terms. This task is significantly harder in comparison to the image classification or object recognition tasks that have been well researched. Starting from 2015 the task has generally been addressed with pipelines composed of a visual encoder and a language model for text generation. During these years, both components have evolved considerably through the exploitation of object regions, attributes, the introduction of multimodal connections, fully attentive approaches, and BERT-like early-fusion strategies. However, regardless of the impressive results, research in image captioning has not reached a conclusive answer yet. This work aims at providing a comprehensive overview of image captioning approaches, from visual encoding and text generation to training strategies, datasets, and evaluation metrics. In this respect, we quantitatively compare many relevant state-of-the-art approaches to identify the most impactful technical innovations in architectures and training strategies.*

*The biggest challenge is most definitely being able to create a description that must capture not only the objects contained in an image, but also express how these objects relate to each other.*

## 1. Introduction

The challenge of captioning an image involves using a visual understanding system and a language model that can produce coherent and syntactically sound sentences to describe the visual content of the picture in natural language.Just recently has neuroscience research made the connection between human vision and language production clear.

Similar to this, artificial intelligence (AI) architectures that can analyse photos and produce words are a relatively new development. Finding the most efficient pipeline to analyse an input image, represent its content, and convert it into a sequence of words by creating links between visual and textual elements while keeping language fluency is the aim of these research endeavors.

The research community has made significant advancements in model design over the past few years, starting with the first deep learning-based proposals that adopted Recurrent Neural Networks (RNNs) fed with global image descriptors, and progressing through attentive approaches, reinforcement learning, and breakthroughs involving Transformers, self-attention, and single-stream BERT-like approaches.

The difficulty of developing appropriate assessment methods and metrics to compare findings with human-generated ground truths has been tackled by the Computer Vision and Natural Language Processing (NLP) communities at the same time. Nonetheless, despite the research and advancements made in these years, picture captioning is still far from being thought of as a problem that has been addressed.

## 2. Dataset

A number of datasets are used for training, testing, and evaluation of the image captioning methods. The datasets differ in various perspectives such as the number of images, the number of captions per image, format of the captions, and image size. Three datasets: Flickr8k, Flickr30k, and MS COCO Dataset are popularly used. In the Flickr8k dataset, each image is associated with five different captions that describe the entities and events depicted in the image that were collected. By associating each image with multiple, independently produced sentences, the dataset captures some of the linguistic variety that can be used to describe the same image. Flickr8k is a good starting dataset as it is small and can be trained easily on low-end laptops/desktops using a CPU.

Our Flick8k dataset structure contains the 8000 images, also Flick8k Text contains the image id along with the 5 captions, training image id's  test image id's.

Word vectors map words to a vector space, where similar words are clustered together and different words are separated. The advantage of using Glove over Word2Vec is that GloVe does not just rely on the local context of words, but it incorporates global word co-occurrence to obtain word vectors.

The basic premise behind Glove is that we can derive semantic relationships between words from the co-occurrence matrix. For our model, we will map all the words in our 38-word long caption to a 200-dimension vector using Glove.

## 3. Literature Survey

There have been several attempts to provide solutions to this problem, including template-based solutions using image classification. However, more recent research has focused on recurrent neural networks. RNNs are already very popular for several natural language processing tasks such as: B. A machine translation that produces a sequence of words. The Image Caption Generator extends the same application by generating word-by-word descriptions of images.

Computer vision reads an image and sees it as a two-dimensional array. Venugopalan thus described captioning as a language translation problem. Previously, language translation was complex and involved several different tasks, but recent research shows that recurrent neural networks can be used to accomplish the task in a highly efficient manner. shown. However, his regular RNN suffers from the vanishing gradient problem, which was important in our application case. The solution to this problem is to use LSTM and GRU. They contain internal mechanisms and logic gates that store information for a long time and pass only useful information.

One of the biggest challenges we faced was choosing a suitable model for our subtitling network. In their work, Tanti categorized generative models into two types: injection architectures and merge architectures. In the former he puts both the tokenized label and the image vector into the RNN block, whereas in the latter he puts the label into the RNN block and merges the output with the image. Experiments show that there is no significant difference in the accuracy of the two models, but we chose the merge architecture due to the simplicity of the design. This reduces hidden states and speeds up training. Images are also not repeatedly passed through her RNN network, which makes better use of RNN memory.

## 4. Experiments

We will tackle this problem using an Encoder-Decoder model. Here our encoder model will combine both the encoded form of the image and the encoded form of the text caption and feed to the decoder.

Our model will treat CNN as the 'image model' and the RNN/LSTM as the 'language model' to encode the text sequences of varying length. The vectors resulting from both the encodings are then merged and processed by a Dense layer to make a final prediction.

We will create a merge architecture in order to keep the image out of the RNN/LSTM and thus be able to train the part of the neural network that handles images and the part that handles language separately, using images and sentences from separate training sets.

In our merge model, a different representation of the image can be combined with the final RNN state before each prediction.

With an Attention mechanism, the image is first divided into n parts, and we compute an image representation of each When the RNN is generating a new word, the attention mechanism is focusing on the relevant part of the image, so the decoder only uses specific parts of the image.

The transformer network employs an encoder-decoder architecture similar to that of an RNN. The main difference is that transformers can receive the input sentence/sequence in parallel, i.e, there is no time step associated with the input, and all the words in the sentence can be passed simultaneously.

## 5. Methodology

We have implemented 3 different techniques to implement Image Captioning.

### 5.1. Merged Encoder - Decoder Architecture

The above diagram is a visual representation of our approach.

The merging of image features with text encodings to a later stage in the architecture is advantageous and can generate better quality captions with smaller layers than the traditional inject architecture (CNN as encoder and RNN as a decoder).

To encode our image features we will make use of transfer learning. There are a lot of models that we can use like VGG-16, InceptionV3, ResNet, etc. We will make use of the inceptionV3 model which has the least number of training parameters in comparison to the others and also outperforms them.

To encode our text sequence we will map every word to a 200-dimensional vector. For this will use a pre-trained Glove model. This mapping will be done in a separate layer after the input layer called the embedding layer.

Using photos and phrases from different training sets, we will build a merge architecture to keep the image out of

the RNN/LSTM and enable us to train the neural network's parts that deal with images and language independently.

A distinct picture representation can be blended with the final RNN state prior to each prediction in our merging model.

The merging of image features with text encodings to a later stage in the architecture is advantageous and can generate better quality captions with smaller layers than the traditional inject architecture (CNN as encoder and RNN as a decoder).

To encode our image features we will make use of transfer learning. There are a lot of models that we can use like VGG-16, InceptionV3, ResNet, etc. We will make use of the inceptionV3 model which has the least number of training parameters in comparison to the others and also outperforms them.

To encode our text sequence we will map every word to a 200-dimensional vector. For this will use a pre-trained Glove model. This mapping will be done in a separate layer after the input layer called the embedding layer.

### 5.2. Attention Mechanism

Rather than compressing an entire image into a static representation, the Attention mechanism allows for salient features to dynamically come to the forefront as and when needed. This is especially important when there is a lot of clutter in an image.

RNNs tend to be computationally expensive to train and evaluate, so in practice, memory is limited to just a few elements. Attention models can help address this problem by selecting the most relevant elements from an input image.

With an Attention mechanism, the image is first divided into n parts, and we compute an image representation of each When the RNN is generating a new word, the attention mechanism is focusing on the relevant part of the image, so the decoder only uses specific parts of the image.

In Bahdanau or Local attention, attention is placed only on a few source positions. As Global attention focuses on all source side words for all target words, it is computationally very expensive. To overcome this deficiency local attention chooses to focus only on a small subset of the hidden states of the encoder per target word.

Local attention first finds an alignment position and then calculates the attention weight in the left and right windows where its position is located and finally weights the context vector. The main advantage of local attention is to reduce the cost of the attention mechanism calculation.

In the calculation, the local attention is not to consider all the words on the source language side, but to predict the position of the source language end to be aligned at the current decoding according to a prediction function and then navigate through the context window, considering only the words within the window.

All hidden states of the encoder and the decoder are used to generate the context vector. The attention mechanism aligns the input and output sequences, with an alignment score parameterized by a feed-forward network. It helps to pay attention to the most relevant information in the source sequence. The model predicts a target word based on the context vectors associated with the source position and the previously generated target words.

To evaluate our captions in reference to the original caption we make use of an evaluation method called BLEU. It is the most widely used evaluation indicator. It is used to analyze the correlation of n-gram between the translation statement to be evaluated and the reference translation statement.

In this article, multiple images are equivalent to multiple source language sentences in the translation. The advantage of BLEU is that the granularity it considers is an n-gram rather than a word, considering longer matching information. The disadvantage of BLEU is that no matter what kind of n-gram is matched, it will be treated the same.

### 5.3. Transformers

The transformer network follows an encoder-decoder architecture that is similar to an RNN, but with a crucial difference. Unlike RNNs, transformers can process input sequences in parallel, without any associated time steps. However, the absence of recurrence and convolution in transformers can lead to a problem where the same word in different sentences can have different meanings. To solve this issue, the transformer network uses position encoding, which encodes information related to a specific position of a token in a sentence using fixed or learned weights.
Similarly, the output embedding of the target German word is fed into the decoder block along with its positional encoding vector.

The encoder block consists of two sub-layers, the first of which is a multi-head self-attention mechanism, while the second is a simple, position-wise fully connected feed-forward network. This allows for the generation of an attention vector for each word, capturing contextual relationships between words in a sentence. Multi-headed attention in the encoder uses self-attention, which enables the model to associate each word in the input with other words.

In addition to the two sub-layers in each encoder layer, the decoder introduces a third sub-layer, performing multi-head attention over the output of the encoder stack. The decoder also utilizes residual connections around each sub-layer, followed by layer normalization. The attention vectors of the German words and English sentences from the encoder are passed into the second multi-head attention, which determines the relatedness of each word vector to each other. This is where the English to German word mapping occurs. The decoder is completed with a linear layer that acts as a classifier, and a softmax function is used to obtain word probabilities.

Transformers are a powerful tool for image captioning, where the aim is to generate natural language descriptions of images. The transformer network learns to encode information about the visual features of the image into a sequence of tokens, which can then be decoded into a caption.

One way to use transformers for image captioning is to treat each image as a sequence of feature vectors, where each vector corresponds to a different spatial location in the image. The encoder block then processes these feature vectors, allowing the model to capture the relationships between different visual features.

Once the encoder has processed the image features, the decoder block can generate a sequence of tokens that describe the image. The decoder is initialized with a special ”start” token, and at each time step, the model generates a probability distribution over the possible next tokens. The model then samples from this distribution to select the next token in the sequence.

The decoder block is similar to the encoder block, but with an additional attention mechanism that allows the model to attend to specific parts of the image while generating the caption. This attention mechanism works by computing a set of attention weights over the image features, which are then used to compute a weighted sum of the features. This allows the model to focus on specific regions of the image while generating the caption.

Overall, transformers have proven to be highly effective for image captioning, achieving state-of-the-art performance on several benchmark datasets. By encoding information about the visual features of the image into a sequence of tokens, transformers are able to generate natural language descriptions that are both accurate and fluent.

### 5.3.1 Positional Encoding

Positional encoding is a technique used in transformer models to inject sequence information into the model. Unlike other sequence models, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), transformers do not take the order of the sequence into account, as they process the entire sequence in parallel. Positional encoding provides a way to encode the position of the tokens in the sequence in a way that the transformer model can understand.

Positional encoding is added to the input embedding of the transformer model. The idea behind positional encoding is to create a vector representation that captures the position of each token in the sequence. This vector is then added to the token’s embedding, which is then passed to the model. The positional encoding is designed in such a way that the cosine and sine functions of different frequencies are added to the input embedding, creating a unique position for each token in the sequence. By doing so, the transformer model can learn to associate each token’s position with its corresponding output.

One important thing to note is that the positional encoding is learned along with the other parameters of the model during training. This means that the model can learn the optimal way to encode the position information of the tokens. By doing so, the transformer model can learn the sequence order and make use of it in its attention mechanism, without relying on the order of the tokens in the input sequence. Overall, the positional encoding is an essential component of the transformer model that allows it to process sequential data accurately and efficiently.

#### 5.3.2 Multi-head attention

Multi-head attention is a crucial component of the Transformer architecture used in natural language processing tasks such as machine translation, question answering, and text summarization. The attention mechanism allows the model to focus on specific parts of the input sequence when generating the output sequence, and multi-head attention extends this by performing attention computations across multiple heads, or subsets, of the input sequence simultaneously.

In multi-head attention, the input sequence is first projected into multiple subspaces, each of which is represented by a separate head. These heads then compute their own attention weights and output sequences, which are concatenated and projected back to the original dimensionality. This allows the model to capture different aspects of the input sequence in parallel, improving both computational efficiency and overall performance.

In addition to improving performance, multi-head attention also offers greater interpretability compared to traditional attention mechanisms. By examining the attention weights generated by each head, researchers can gain insights into the specific parts of the input sequence that are most relevant for generating the output sequence. This can be particularly useful for tasks such as text summarization, where it is important to identify the most important information in the input text. Overall, multi-head attention is a powerful tool for natural language processing tasks that require attention mechanisms, providing both improved performance and interpretability.

#### 5.3.3 Multi-head attention

The encoder-decoder layer in transformers is a crucial component that enables the model to generate high-quality sequences. The encoder takes the input sequence and generates a set of feature vectors, where each vector contains information about a specific element in the input sequence. The decoder then takes the encoded vectors and generates the output sequence one element at a time. To ensure that the decoder generates semantically coherent sequences, it uses the encoded vectors as a context vector to generate each output element. This approach allows the model to capture complex dependencies between input and output sequences, making it suitable for a wide range of natural language processing tasks.

The encoder-decoder layer consists of two parts, the encoder and the decoder. The encoder is typically composed of several identical layers, each containing two sub-layers. The first sub-layer is a multi-head self-attention mechanism that allows the model to capture dependencies between different elements of the input sequence. The second sub-layer is a position-wise feed-forward network that applies a non-linear transformation to each element of the sequence independently. The output of the encoder is a set of feature vectors that contain information about each element of the input sequence.

The decoder, on the other hand, is similar to the encoder, but with an additional third sub-layer that performs multi-head attention over the encoded vectors. The first sub-layer in the decoder is a masked multi-head self-attention mechanism that ensures the model only attends to previous elements in the output sequence. The second sub-layer is also a position-wise feed-forward network. Finally, the third sub-layer is a multi-head attention mechanism that allows the model to attend to the encoded vectors. The output of the decoder is a set of feature vectors that contain information about each element of the output sequence. By using the encoded vectors as context vectors, the decoder can generate semantically coherent output sequences that depend on the input sequence.

## 6. Evaluation Techniques

We have implemented 3 Evaluation Techniques for our Image Caption Generator.

### 6.1. Greedy Search

It is a local search algorithm that always chooses the best available option at each step, without considering the long-term consequences of that choice. The algorithm works by starting with an initial solution and then iteratively improving it by choosing the best possible option at each step. Greedy search is easy to implement and computationally efficient, making it a popular choice for solving a wide range of problems in various fields, including computer science, mathematics, and engineering.

It can get stuck in local optima, where a suboptimal solution is chosen early on and then further improvements become impossible. Greedy search also does not guarantee that the final solution will be globally optimal, as it does not explore all possible options before making a decision.

As the model generates a 1660 long vector with a probability distribution across all the words in the vocabulary we greedily pick the word with the highest probability to get the next word prediction.

Although greedy search can provide quick solutions to problems, it may not always be the best approach, especially for complex problems that require a global optimization solution. To overcome the limitations of greedy search, various modifications have been proposed, including randomized greedy search, beam search, and simulated annealing.

Randomized greedy search introduces a randomized element to the greedy algorithm, which helps to explore different solutions and avoid getting stuck in a local optima. Beam search is another modification that explores multiple paths simultaneously, keeping track of the top-k candidates

at each step. This helps to avoid discarding potentially good solutions early on and allows for a more extensive search space. Simulated annealing is a probabilistic optimization technique that allows the algorithm to escape from local optima by accepting worse solutions with a certain probability.

In image captioning, the use of greedy search has its limitations, as it only considers the highest probability word at each step, which may lead to incorrect predictions. For example, the model may predict "dog" instead of "cat" because "dog" has a slightly higher probability, even though "cat" may be the correct word to use in the context of the image. To improve the accuracy of image captioning, various modifications to the greedy search algorithm have been proposed, including incorporating beam search or using a combination of greedy search and beam search.

Despite its limitations, greedy search remains a popular choice for solving many problems, and its simplicity and efficiency make it an attractive option for real-time applications. However, when dealing with complex problems or when the optimal solution is critical, it is essential to consider other optimization techniques that may provide better results.

### 6.2. Beam Search

This algorithm is an extension of the greedy search algorithm and works by maintaining a set of the most promising candidate solutions, called the "beam," at each step of the search. Unlike greedy search, beam search considers a limited set of possible solutions at each step, which helps to reduce the computational complexity of the algorithm. This also allows beam search to explore a larger portion of the search space, potentially leading to a better global solution.

Beam Search is where we take top k predictions, feed them again in the model and then sort them using the probabilities returned by the model. So, the list will always contain the top k predictions and we take the one with the highest probability and go through it till we encounter 'endseq' or reach the maximum caption length.

One of the limitations of beam search is that it may not necessarily result in the best possible solution, as it is dependent on the number of candidates chosen at each step. Setting the beam size too low can lead to a suboptimal solution, while setting it too high can increase computational complexity and memory usage. Additionally, beam search may generate redundant or repetitive phrases in the output, as it does not explicitly consider the coherence of the sentence as a whole.

To address these limitations, various modifications to beam search have been proposed, such as length normalization, diverse beam search, and nucleus sampling. Length normalization adjusts the score of each candidate solution based on its length, favoring shorter solutions. Diverse beam search encourages the selection of diverse solutions by penalizing candidates that are similar to previously chosen solutions. Nucleus sampling samples from a subset of the most probable words instead of considering all possible words, reducing the chance of generating low-quality or uncommon words. These modifications can improve the overall performance of the beam search algorithm in generating high-quality image captions.

### 6.3. BLEU

BLEU (Bilingual Evaluation Understudy) is a widely used metric for evaluating the quality of machine translation outputs against human references. BLEU measures the n-gram overlap between the machine-generated translation and the reference translations, where n-grams are sequences of n words. It also considers the length of the translation and applies a brevity penalty to penalize translations that are too short compared to the reference translations. BLEU scores range from 0 to 1, with higher scores indicating better translation quality.

BLEU has been widely adopted as an evaluation metric in machine translation research and is used by many machine translation systems to optimize their models. It has also been extended in several ways to improve its effectiveness, such as adding smoothing techniques to handle sparse n-gram counts and modifying the n-gram penalty to better reflect human judgment. Despite its popularity, however, BLEU has been criticized for its limitations, such as its inability to capture semantic meaning and its dependence on reference translations, which may not always be available or representative of human preferences.

BLEU is based on the assumption that a good translation should have a high degree of overlap with the reference translation. However, this assumption may not always hold true, especially in cases where the reference translations are not comprehensive or may contain errors. Additionally, BLEU does not take into account the grammaticality or fluency of the generated translation, which are also important aspects of translation quality.

To address some of these limitations, other metrics have been proposed, such as METEOR, which incorporates semantic information and matches between phrases, and TER (Translation Edit Rate), which measures the edit distance between the generated and reference translations. Despite the availability of alternative metrics, BLEU remains widely used and serves as a standard benchmark for evaluating machine translation systems.

In recent years, there has been increasing interest in developing more advanced evaluation metrics that can better capture the nuances of human language and provide a more comprehensive evaluation of translation quality. One such metric is the human evaluation, where human judges rate the quality of translations based on various criteria, such as fluency, accuracy, and naturalness. Human evaluation is of-

ten considered the most reliable method of evaluating translation quality, but it is also expensive and time-consuming. Nonetheless, it is likely to play an increasingly important role in the development of machine translation systems, as researchers seek to bridge the gap between machine and human translation capabilities.

# 7. Results

## 7.1. Merge Encoder - Decoder

We will test our model on the merge encoder - decoder architecture and view its performance.

First, we will take a look at the example dog image. The caption for the image is 'A black dog and a brown dog in the snow'. Let's see how our model compares.

```
Greedy Search: two dogs are running through the snow
Beam Search, K = 3: white dog is running through the snow
Beam Search, K = 5: brown and white dog is running through the snow
Beam Search, K = 7: brown and white dog is running through the snow
Beam Search, K = 10: brown and white dog is running through the snow
```

You can see that our model was able to identify two dogs in the snow. But at the same time, it misclassified the black dog as a white dog. Nevertheless, it was able to form a proper sentence to describe the image as a human would.

Next, we will take a look at the another example image.Let's see how our model compares.

```
Greedy: a group of people are standing around a figure in a city
Beam Search, K = 3: a group of people are standing in front of a building
Beam Search, K = 5: a group of people are standing in front of a building
Beam Search, K = 7: a group of people are standing in front of a building
```

Here we can see that we accurately described what was happening in the image. You will also notice the captions generated are much better using Beam Search than Greedy Search.

## 7.2. Attention Mechanism

We will test our model on the attention mechanism architecture and view its performance.

You can see even though our caption is quite different from the real caption, it is still very accurate. It was able to identify the yellow shirt of the woman and her hands in the pocket.

## 7.3. Transformers

We will test our model on the transformer architecture and view its performance.

```
BLEU-1 score: 38.94003915357025
BLEU-2 score: 55.06953149031838
BLEU-3 score: 63.25828023338613
BLEU-4 score: 65.48907866815301
Real Caption: dog leaps to catch ball
Predicted Caption: dog catches ball outside
<matplotlib.image.AxesImage at 0x7fa3706d5ef0>
```

Now running transformers architecture on the same data set we can see an improved performance on our image captioning using BLEU evaluation Technique. We have run BLEU 1-4 indicators to evaluate the best text for the image

and we get a best score of 65 for the above example.

```
BLEU-1 score: 33.33333333333333
BLEU-2 score: 57.735026918962575
BLEU-3 score: 71.92230933248644
BLEU-4 score: 75.98356856515926
Real Caption: dog standing outside is looking at the camera
Predicted Caption: tan and white dog is running through the leaves
<matplotlib.image.AxesImage at 0x7fa140259fd0>
```

Similarly, for another example we see transformer architecture outperforming other methods

## 8. Conclusion

In conclusion, the image caption generator using a merged encoder-decoder and attention mechanism is a powerful technique for generating natural language descriptions of images. By combining the strengths of both encoder-decoder models and attention mechanisms, this approach is able to accurately capture the key features of an image and produce a semantically coherent and grammatically correct caption. Furthermore, the attention mechanism allows the model to focus on the most relevant parts of the image when generating the caption, leading to more accurate and descriptive captions.

While this approach has demonstrated impressive results, there is still room for improvement. One area for future research is exploring different ways of merging the encoder and decoder models, such as using a hierarchical structure or incorporating additional information sources. Additionally, improvements can be made to the attention mechanism, such as incorporating visual or contextual cues to guide the attention process. Overall, the image caption generator using merged encoder-decoder and attention is a promising approach that has the potential to revolutionize the field of computer vision and natural language processing.

In addition to the merged encoder-decoder and attention mechanism, the use of transformers has also shown to have a significant impact on the performance of image caption generators. By leveraging the self-attention mechanism in transformers, the model is able to capture long-range dependencies and contextual relationships between words more effectively, leading to more coherent and fluent captions.

The use of transformers in image captioning has also opened up new avenues for research, such as exploring pre-training techniques and incorporating multi modal inputs such as audio and video. Overall, the incorporation of transformers in image captioning represents a promising direction for advancing the state-of-the-art in this field, and holds great potential for improving the accuracy and quality of image caption generation.

## References

1. ”Every Picture Tells a Story: Generating Sentences from Images.” Computer Vision ECCV (2016) by Farhadi, Ali, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hocken-maier, and David Forsyth.

2. Automatic Caption Generation for News Images by Yansong Feng, and Mirella Lapata, IEEE (2013).

3. Image Caption Generator Based on Deep Neural Networks by Jianhui Chen, Wenqiang Dong and Minchen Li, ACM (2014).

4. Show and Tell: A Neural Image Caption Generator by Oriol Vinyal, Alexander Toshev, Samy Bengio, Dumitru Erhan, IEEE (2015).

5. Image2Text: A Multimodal Caption Generator by Chang Liu, Changhu Wang, Fuchun Sun, Yong Rui, ACM (2016).