# VISUAL SPEAKER

## Queen's International Innovation Challenge

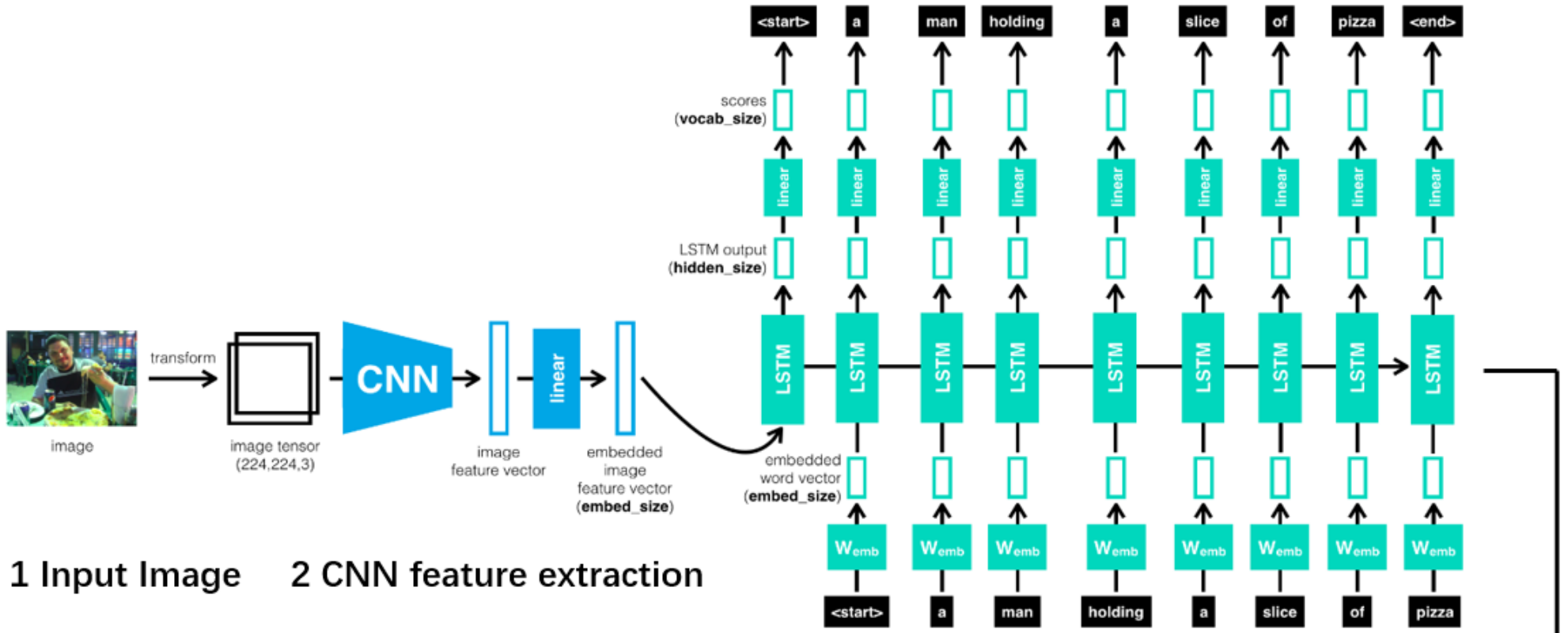
## Smarter Cities & Better Future

MIE 1624 Group 15

# Objective: Bring convenience to the blind

- **285 million** people are visually impaired worldwide, including over **39 million** blind people
- **Vision-to-Language** problems have always been challenging
- Our project: fusing both **image captioning prototype** and a **text-to-speech synthesizer**

# Methodology: Architecture of the Model



# Methodology: Mathematical Interpretation

Model

**Goal**: $S = f(I)$ ← RBG image

Target Description

**Maximize**

$$\theta^* = \arg max_\theta \sum_{(I,S)} \log p(S|I;\theta)$$

Parameters

**Chain Rule**

Preceding Words

$$\log p(S|I) = \sum_{t=0}^{N} \log p(S_t|I, S_0, \cdots, S_{t-1})$$

**CNN**: the visual feature of RGB image is embedded to as the input to the RNN

$$\mu = W_\mu \left(CNN(I)\right)$$

Embedding of visual feature

**RNN (LSTM)**: represented each word by a one-hot vector $S_t$

$$x_t = W_t S_t, t \in \{0 \cdots N-1\}$$

Word embedding parameter

Decode the features $\mu$, $x_t$ and internal hidden parameter $h_t$ into a probability

$$p_{t+1} = LSTM(\mu, x_t, h_t), t \in \{0 \cdots N-1\}$$

# EXPERIMENTAL IMPLEMENTATION

## Dataset

**MSCOCO dataset** provided by Microsoft

**80K training** images,
**40K validation** images and
**40K test** images.


## Vocabulary

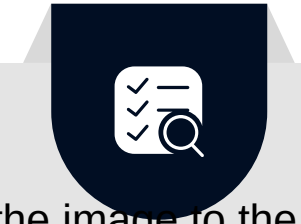
**6325 unique words** from the training captions that appear at least 10 times

Less frequently words are replaced by a special **OUT-OF-VOCABULARY** token

```
A cat sitting in front of a delicious chocolate donut.
A cat looking at a donut on a plate.
A cat is looking over a chocolate covered donut on a plate.
A cat stares at a chocolate topped donut, with the caption reading, "DONUT WANT."
Cat looks at donut with words "donut want" along bottom
```

Start and the end of the caption is marked with a special **START** and **STOP** token

# EXPERIMENTAL IMPLEMENTATION



- Re-scale the image to the dimension of 224*224 in RGB channel.
- Implement the pretrained model **ResNet50** from PyTorch
- Extract feature representations of the image from last convolution layer of ResNet50
- Use linear mapping to map the features to the same shape of embedding features of RNN part.

**CNN encoder**

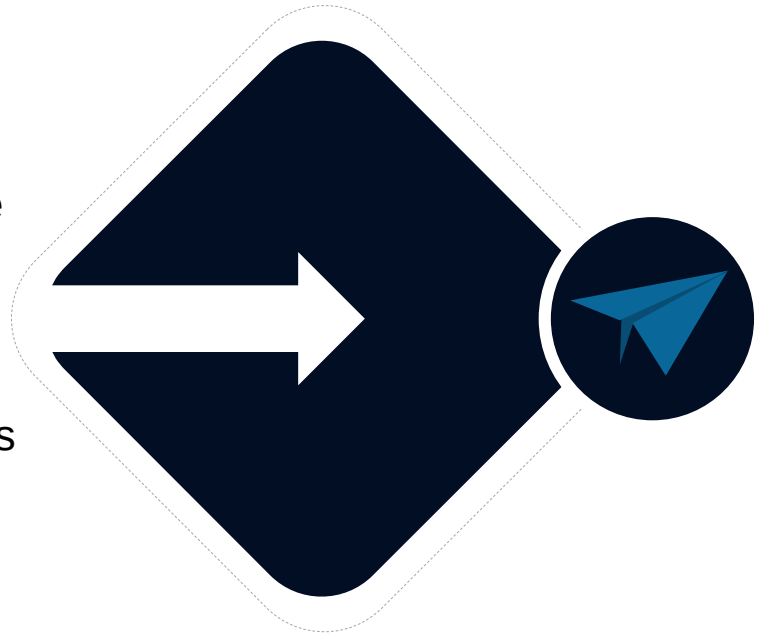
- Based on **LSTM cell-model**
- Obtain feature representation of image which is used as the first input to LSTM
- Obtain the score vector of each word with **hidden layers** and **final linear mapping**
- Add the **soft-max layer** to get the probability vector of each word
- Use **cross-entropy loss** to construct objective function

**RNN decoder**

# EXPERIMENTAL IMPLEMENTATION

## Training

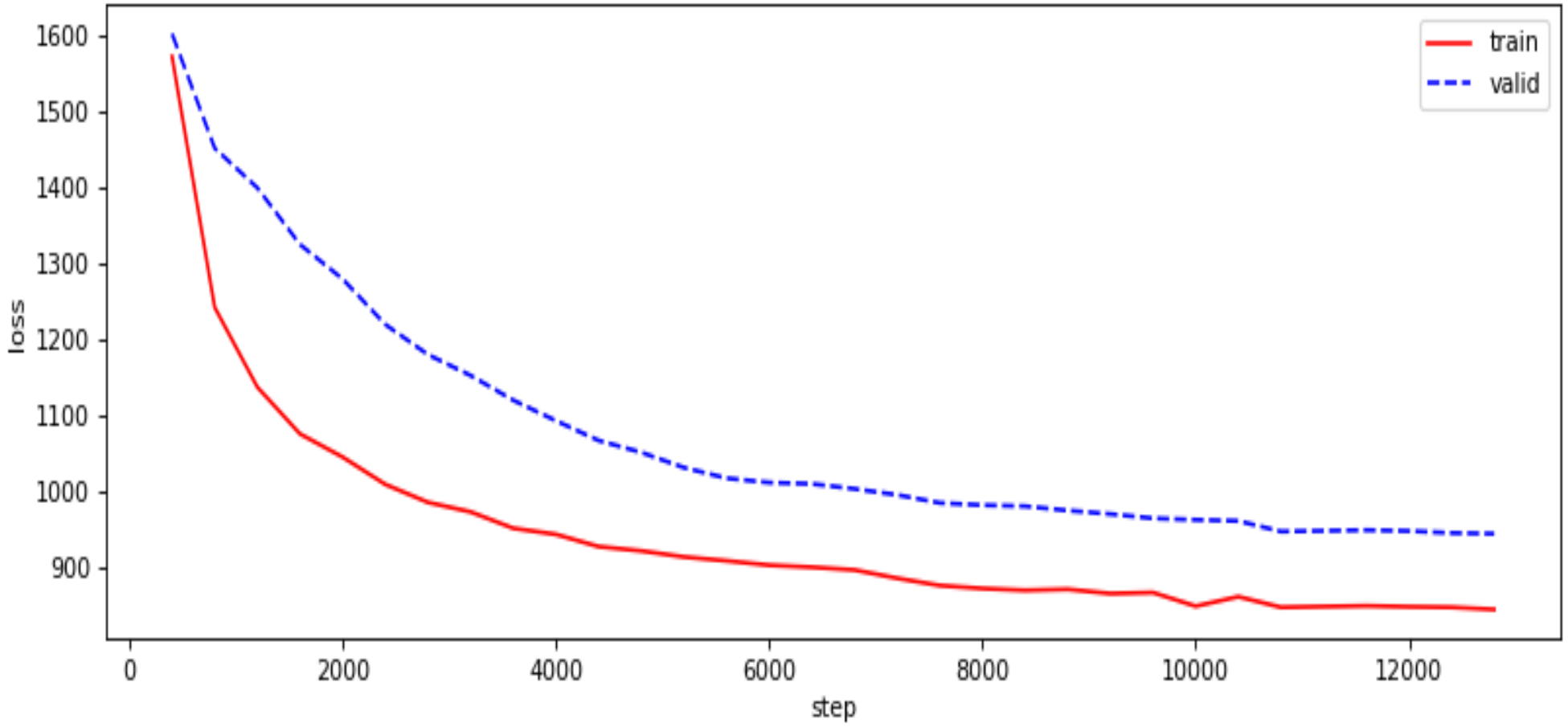
- The neural network is trained using the **mini-patch stochastic gradient descent (SGD)** method.
- The base learning rate is **0.001**. The learning rate drops 50% in every **20,000** iterations.
- Model is selected based on total loss value applied on validation set.
- Implement model with PyTorch framework and train it on **GTX TITAN X 12GB**
- Use metrics BLEU-4 to quantitatively evaluate performance.



## Inference & Web structure

- Feed images to the pre-trained model
- RNN generates word embedding probability vector which is continually fed to RNN to generate next word.
- RNN decoder will generate captions iteratively until it meets STOP token.
- Build a web application in python using Flask framework
- Use IBM Text2Speech API to generate audio file

# Result: Training and Validation Losses



# Result: Positive Result

Text: a living room filled with furniture and a piano .

Text: a bathroom with a shower , toilet , and sink .

Text: a dog sitting on a desk next to a computer monitor .

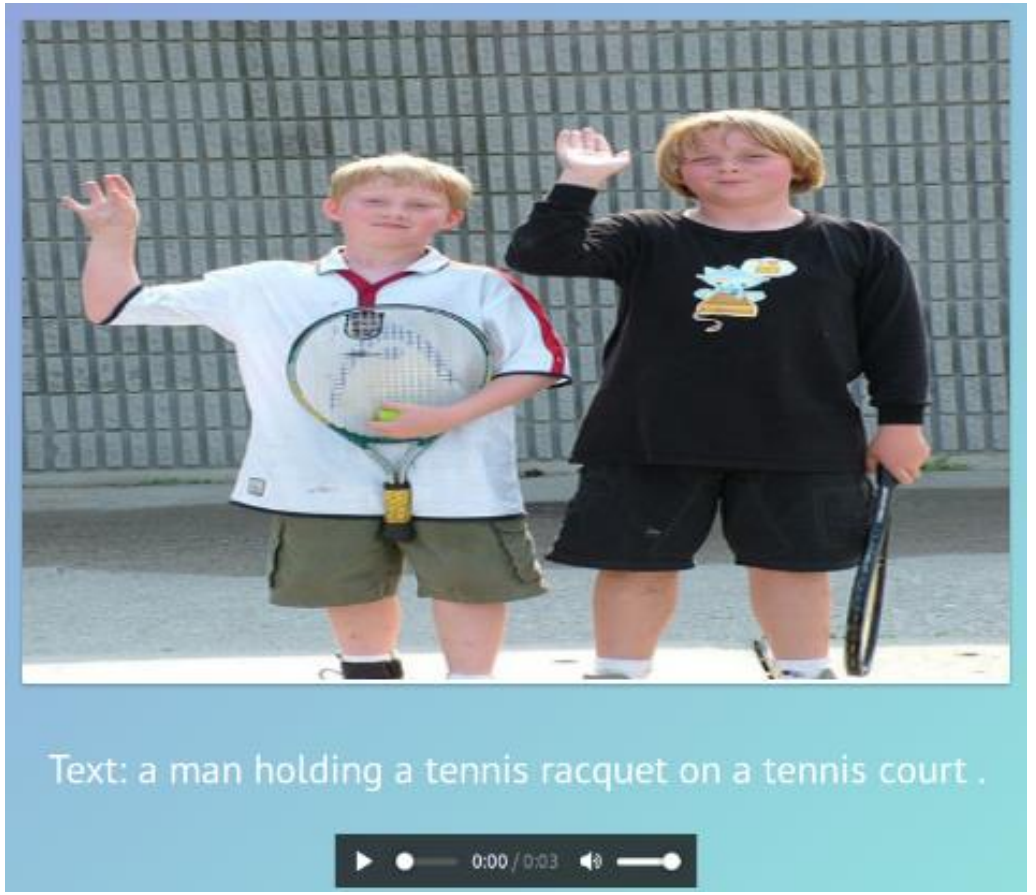
# Result: Positive Result

Text: a red train traveling down train tracks near a forest .

Text: a group of people walking down a city street .

Text: a bus is parked on the side of the road .

# Problem Existed

# Thanks