SPARK+AI SUMMIT 2020

Organized by databricks

Text Extraction from Product Images

Rajesh Shreedhar Bhat

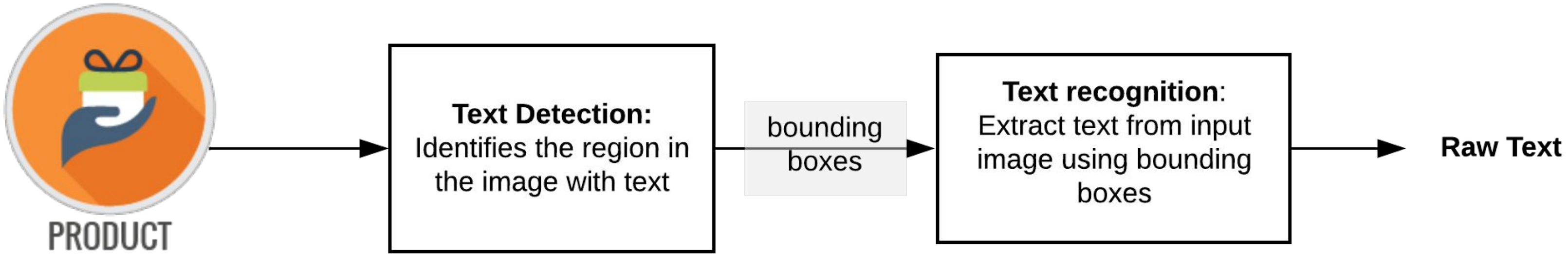
Data Scientist @Walmart Labs, Bengaluru

MS CS @ASU, Kaggle Competitions Expert

Agenda

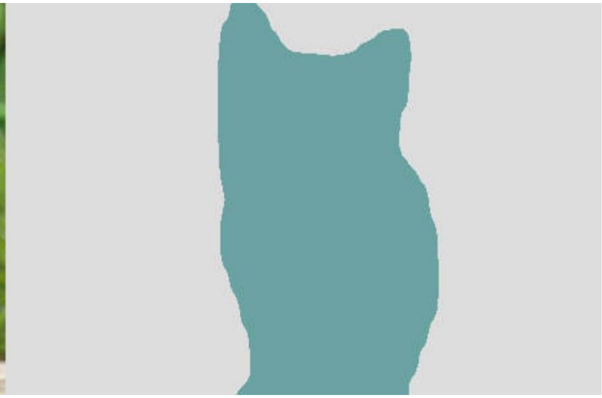
- Intro to Text Extraction
- **T**ext **D**etection(TD)
- TD Model Architecture
- Training data generation
- **T**ext **R**ecognition(TR) training data preparation
- CRNN-CTC model for TR
- Receptive Fields
- CTC decoder and loss
- TR Training phase
- Other Advanced Techniques
- Questions ?

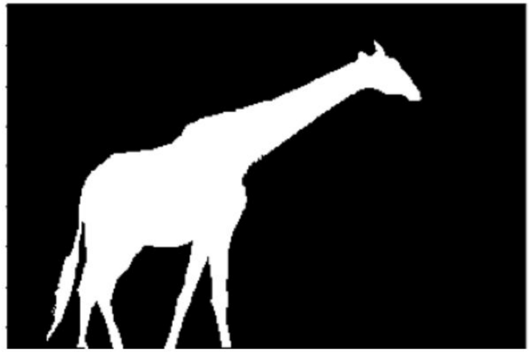
Introduction : Text Extraction

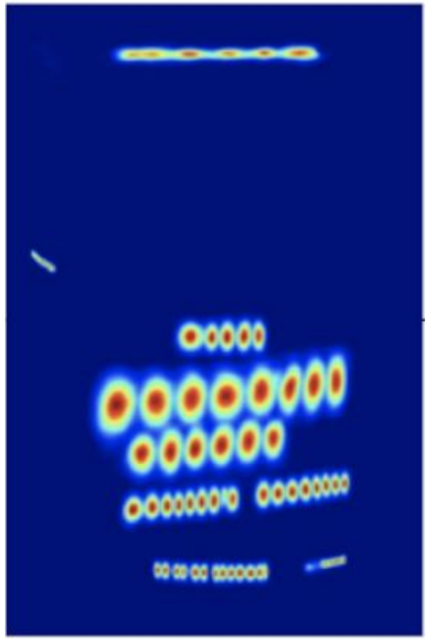


Text Detection

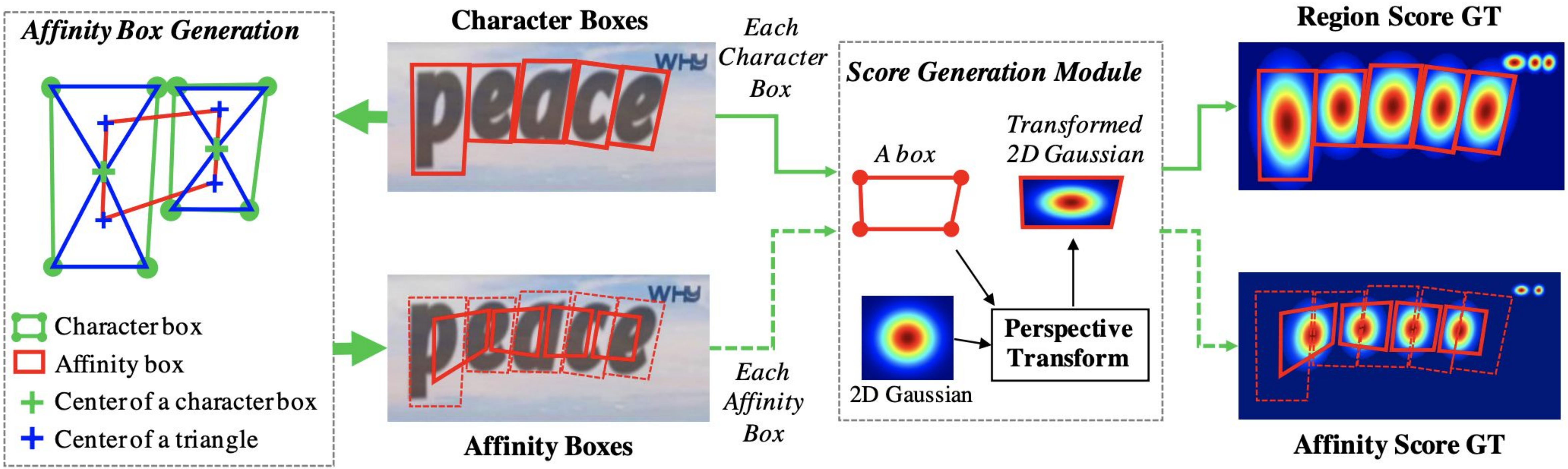
Image Segmentation - Input & Ground Truth



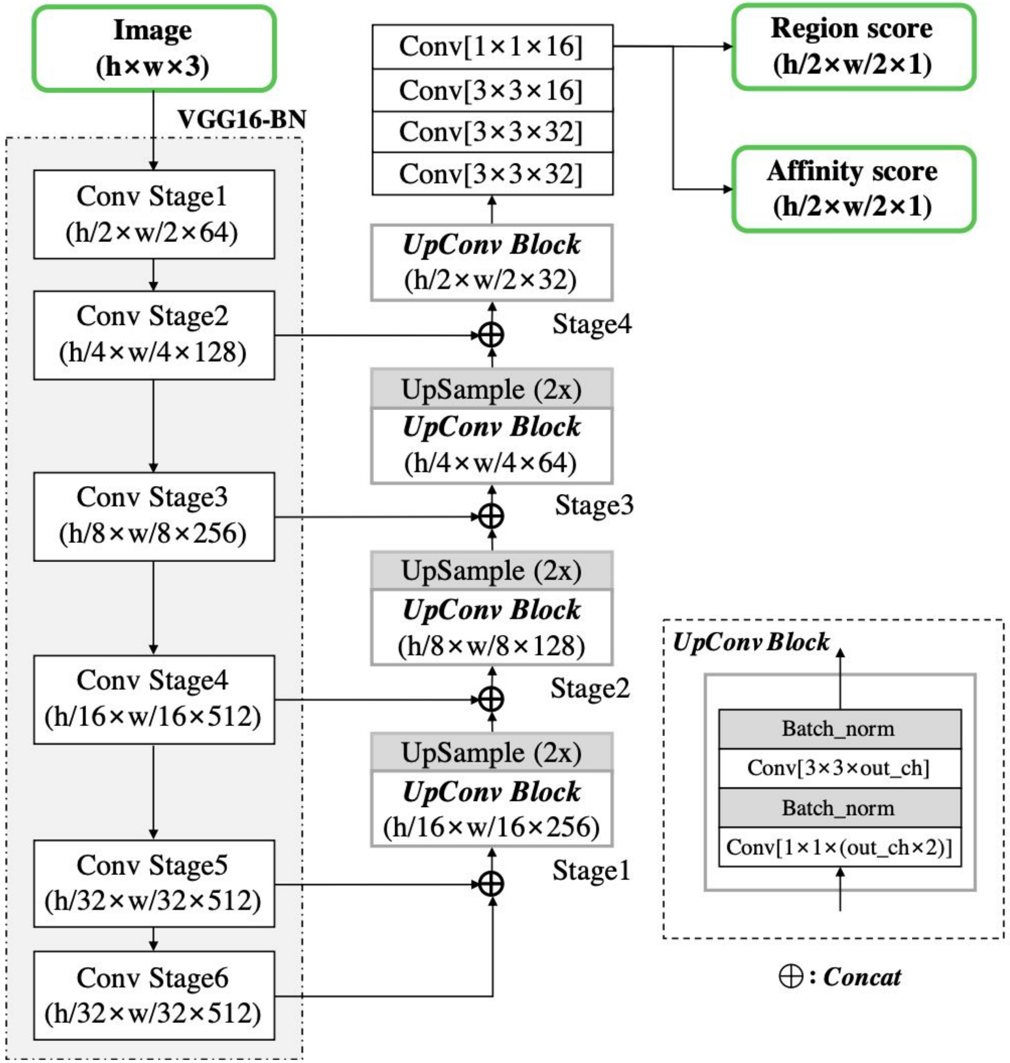
Ground Truth Label Generation



Text Detection – Model architecture

- VGG16 – BN as the backbone
- Model has skip connection in decoder part which is similar to U-Nets.
- Output :
  - Region score
  - Affinity score - grouping characters

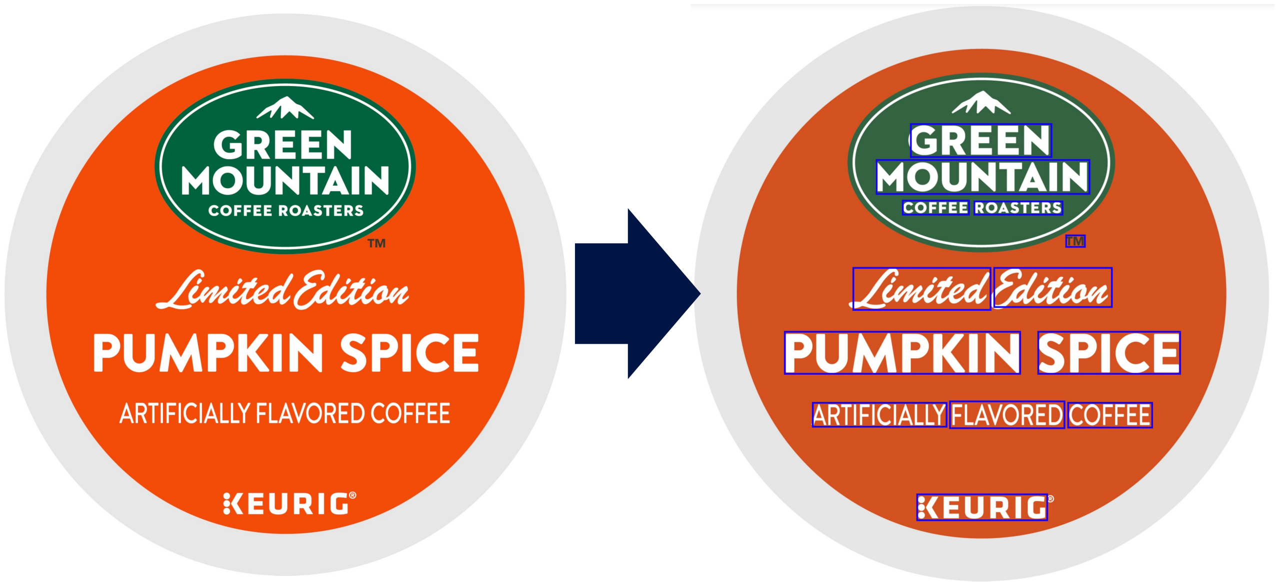
Ref: Baek, Youngmin, et al. "**C**haracter **R**egion **A**wareness for **T**ext detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.



Sample Output

Region Score | Affinity Score | Region Score | Affinity Score

Sample Output..



Text Recognition

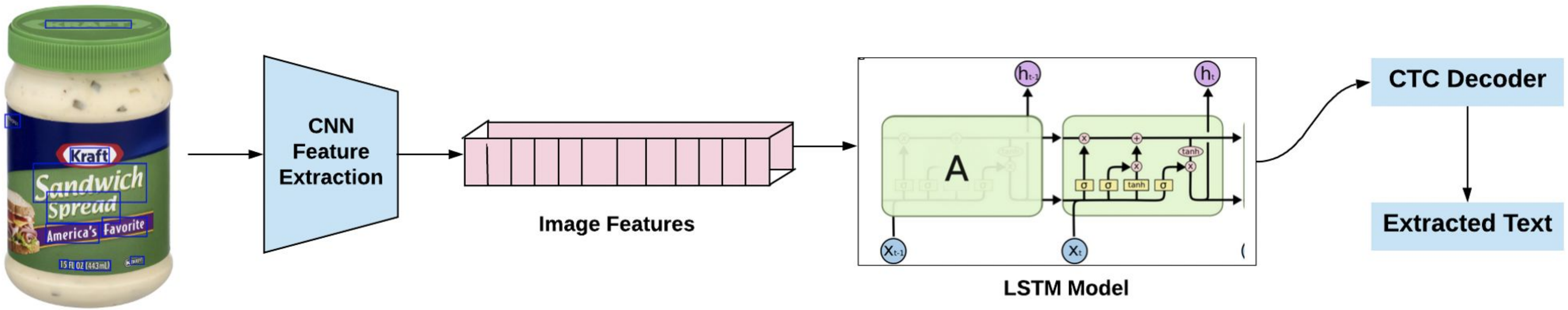
Text Recognition – Training Data Preparation

SynthText: image generation engine for building a large annotated dataset.

15 million images generated with different **font styles, size, color** & **varying backgrounds** using product descriptions + open source datasets

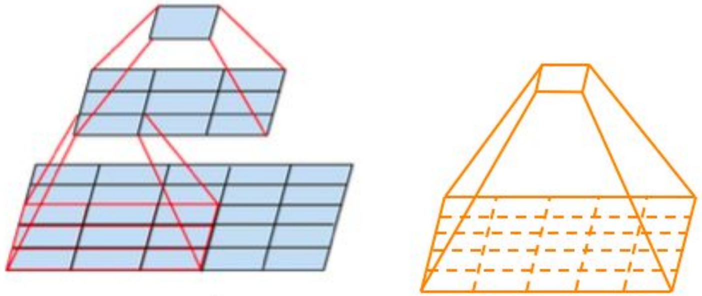
Vocabulary: 92 characters
Includes capital + small letters, numbers and special symbols

Text Recognition CRNN CTC model



CNN - Receptive Fields

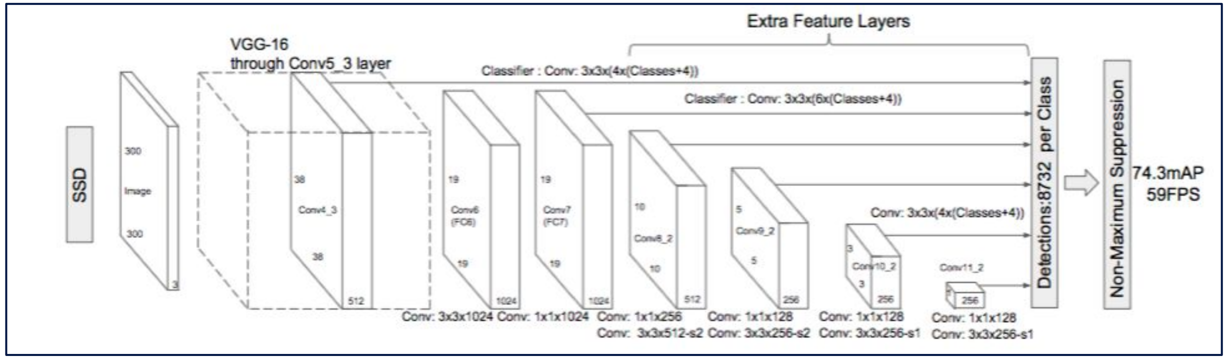
- Receptive field is defined as the region in the input image/space that a particular CNN's feature is looking at.



two successive
3x3 convolutions

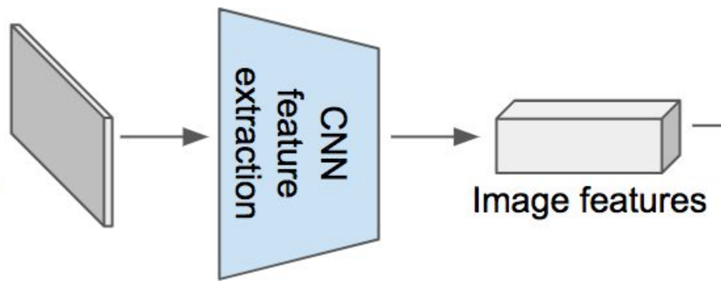
$$n_{out} = \left\lfloor \frac{n_{in} + 2p - k}{s} \right\rfloor + 1$$

Usage of intermediate layer features in SSD's in Object detection tasks.



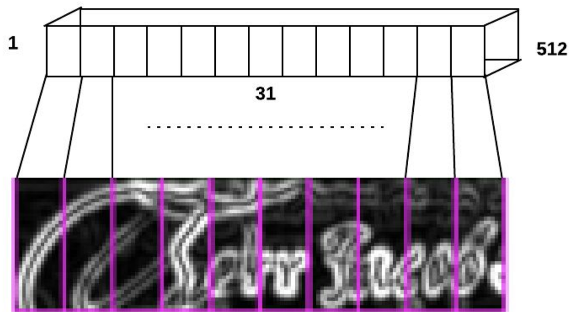
CNN features to LSTM

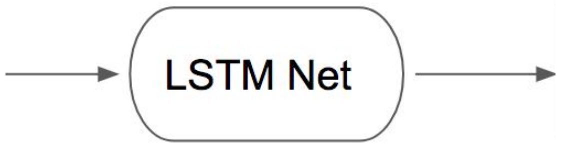
Input Image
(None, 128 * 32 * 1)



Feature map
(None, 1, 31, 512)

Feature map
(None, 1, 31, 512)





SoftMax probabilities for every time step (i.e. 31), over the vocabulary.

Ground Truth and Output of TR task

Input Image

Ground Truth

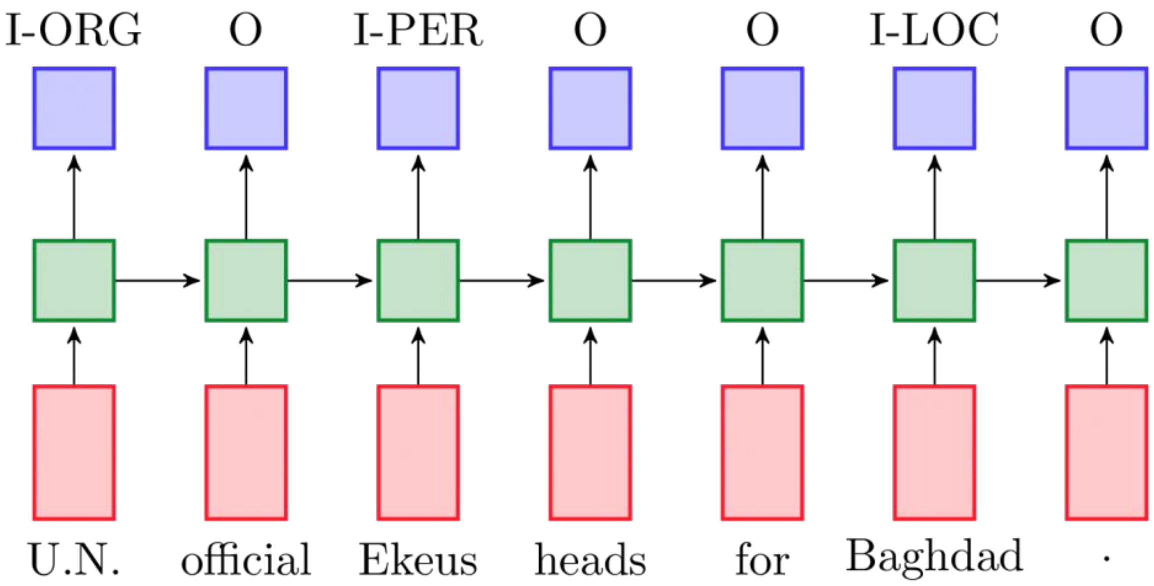
Hello

Output from LSTM model for 31 timesteps ..

Time step	t1	t2	t3	t4	t5	...	....	t27	t28	t29	t30	t31
Prediction	H	H	H	e	e	...	...	l	o	o	o	o

Length of Ground truth is **5** which is **not equal** to length of prediction i.e **31**

How to calculate the loss?

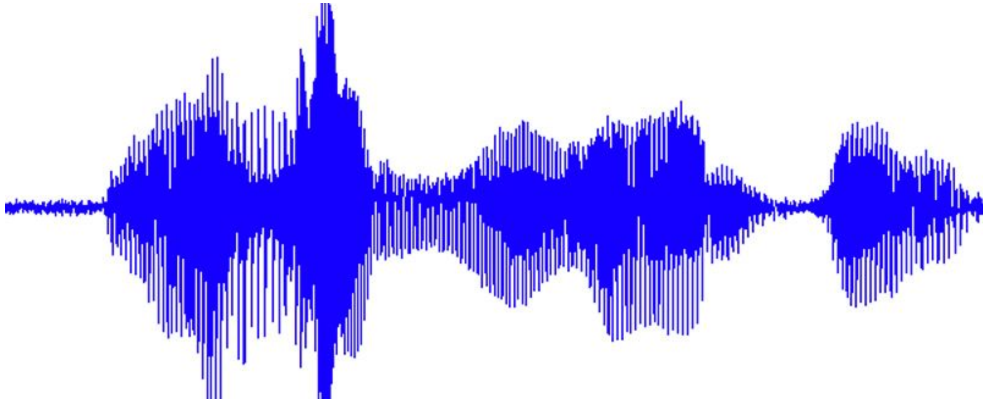


NER model – loss: categorical cross entropy

- Do we have labels for every time steps of LSTM model in CRNN setting ?
- Can we use cross entropy loss?

Answer is: NO!!

Mapping of Input to Output



Corresponding Text → Hello

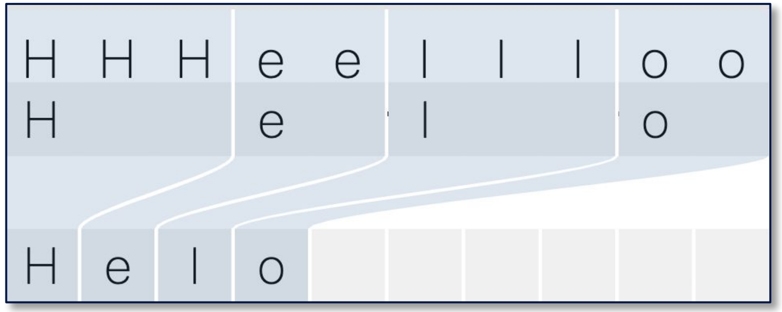
Corresponding Text → Hello

Can we manually align each character to its location in the audio/image?

Yes!! But lot of manual effort is needed in creating training data.

CTC to rescue

With just mapping from image to text and not worrying about alignment of each character to the location in input image, one should be able to train the network.



Merge repeats

Merge repeats
Drop blank character

Connectionist Temporal Connection (CTC) Loss

- Ground truth for an image 

—> **AB**

- Vocabulary is **{ A, B, - }**

- Let's say we have predictions for **3-time** steps from LSTM network (SoftMax probabilities over vocabulary at **t1, t2, t3**)

- Given that we use CTC decode operation discussed earlier, in which scenarios we can say output from the model is correct??

CTC loss continued ..

Ground Truth : AB

t1	t2	t3
A	B	B
A	A	B
-	A	B
A	-	B
A	B	-

- Merge repeats
- Drop blank character

AB

SoftMax probabilities

	t1	t2	t3
A	0.8	0.7	0.1
B	0.1	0.1	0.8
-	0.1	0.2	0.1

Score for one path: **AAB** = (0.8 * 0.7 * 0.8) and similarly for other paths.

Probability of getting GT **AB**: = P(ABB) + P(AAB) + P(-AB) + P(A-B) + P(AB-)

Loss : - log(Probability of getting GT)

CTC loss perfect match

Ground Truth : A

t1	t2	t3
A	-	-
-	A	-
-	-	A
-	A	A
A	A	-
A	-	A
A	A	A

- Merge repeats
- Drop blank character

A

- SoftMax probabilities

	t1	t2	t3
A	1	0	0
B	0	0	0
-	0	1	1

Score for one path: **A- -** = (1 * 1 * 1) and similarly for other paths.

Probability of getting ground truth **A**: = P(A—) + P(-A-) + P(—A) + P(-AA) + P(AA-) + P(A-A) + P(AAA)

Loss : - log(Probability of getting GT) = 0

CTC loss perfect mismatch

Ground Truth : A

t1	t2	t3
A	-	-
-	A	-
-	-	A
-	A	A
A	A	-
A	-	A
A	A	A

- Merge repeats
- Drop blank character

A

SoftMax probabilities

	t1	t2	t3
A	0	0	0
B	1	1	1
-	0	0	0

Score for one path: **A - -** = (0 * 0 * 0) and similarly for other paths.

Probability of getting ground truth **A**: = P(A—) + P(-A-) + P(—A) + P(-AA) + P(AA-) + P(A-A) + P(AAA)

Loss : - log(Probability of getting GT) = **tends to infinity !!**

Model Architecture & CTC loss in TF

```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 32, 128, 1)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 128, 64)       640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 16, 64, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 64, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 8, 32, 128)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 32, 256)        295168
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 8, 32, 256)        590080
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 4, 32, 256)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 4, 32, 512)        1180160
_________________________________________________________________
batch_normalization_1 (Batch (None, 4, 32, 512)        2048
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 4, 32, 512)        2359808
_________________________________________________________________
batch_normalization_2 (Batch (None, 4, 32, 512)        2048
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 2, 32, 512)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 1, 31, 512)        1049088
_________________________________________________________________
lambda_1 (Lambda)            (None, 31, 512)           0
_________________________________________________________________
bidirectional_1 (Bidirection (None, 31, 256)           657408
_________________________________________________________________
bidirectional_2 (Bidirection (None, 31, 256)           395264
_________________________________________________________________
dense_1 (Dense)              (None, 31, 93)            23901
=================================================================
Total params: 6,629,469
Trainable params: 6,627,421
Non-trainable params: 2,048
```

```
tf.keras.backend.ctc_batch_cost(
    y_true, y_pred, input_length, label_length
)
```

Arguments	
`y_true`	tensor `(samples, max_string_length)` containing the truth labels.
`y_pred`	tensor `(samples, time_steps, num_categories)` containing the prediction, or output of the softmax.
`input_length`	tensor `(samples, 1)` containing the sequence length for each batch item in `y_pred`.
`label_length`	tensor `(samples, 1)` containing the sequence length for each batch item in `y_true`.

Returns
Tensor with shape (samples,1) containing the CTC loss of each element.

Training Phase

- **15 million images ~ 690 GB when loaded into memory!!** Given that on an average images are of the shape **(128 * 32 * 3)** and **dtype is float32.**

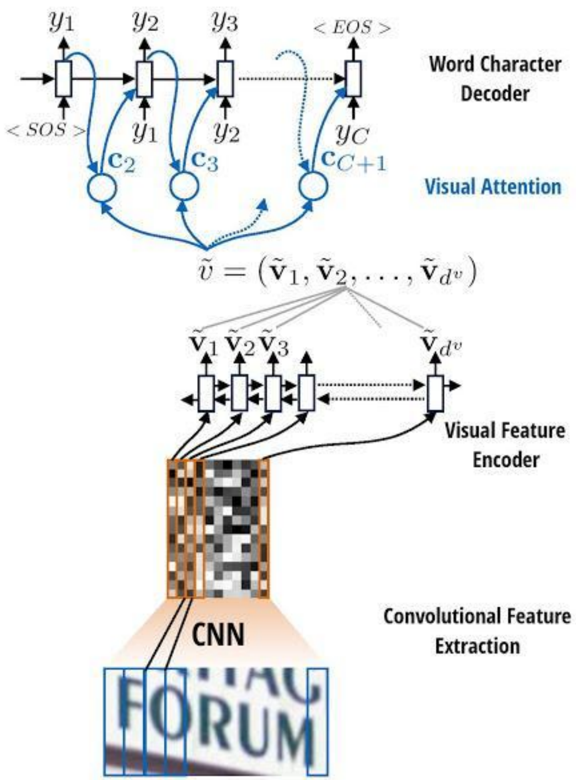
- Usage of Python Generators to load only single batch in memory.

- Reducing the training time by using workers, max_queue_size & multi-processing in .fit_generator in Keras.

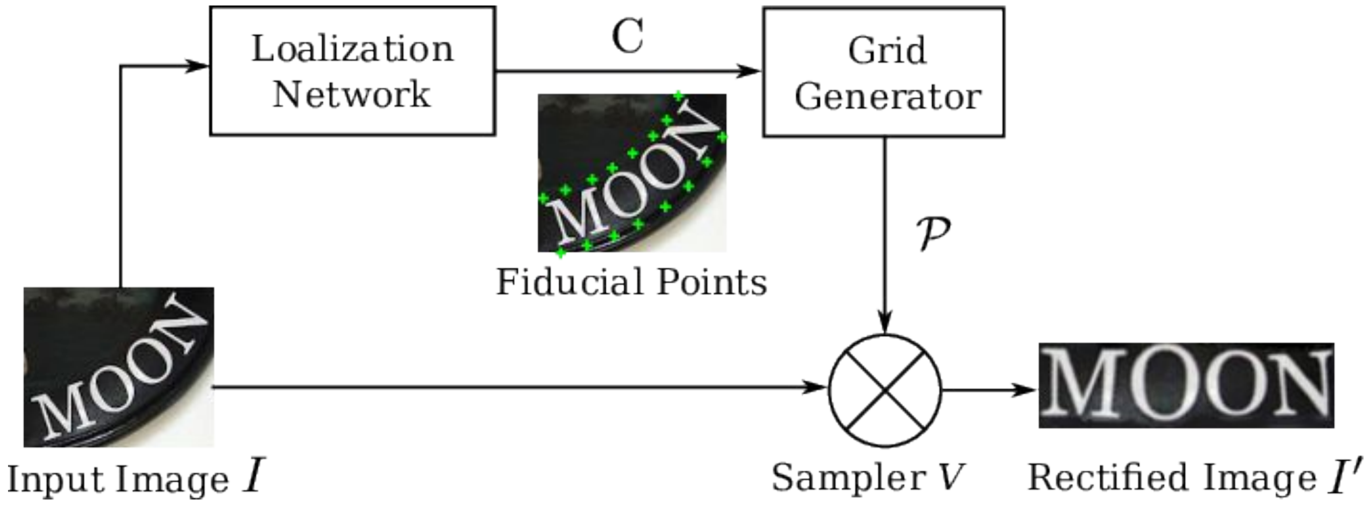
- Training time ~ 2 hours for single epoch on single P100 GPU machine and prediction time ~1 sec for batch of 2048 images.

Other Advanced Techniques

Attention - OCR



Spatial Transformer Network – before text recognition



Ref:[*Jaderberg, Max, Karen Simonyan, and Andrew Zisserman. "Spatial transformer networks." Advances in neural information processing systems. 2015.*]

Code + PPT

https://github.com/rajesh-bhat/spark-ai-summit-2020-text-extraction

Questions ??

rsbhat@asu.edu

https://www.linkedin.com/in/rajeshshreedhar