OCR with MXNet Gluon

Work of Jonathan Chung (& Thomas Delteil)

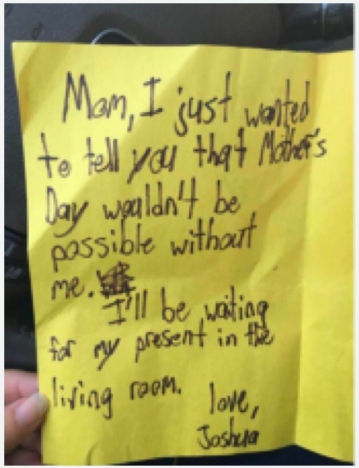
Presented by Thom Lane

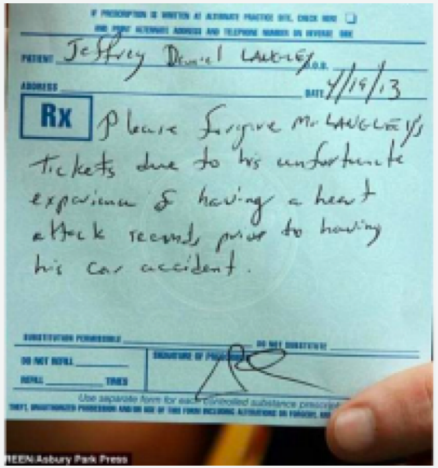
AWS MXNet Applications

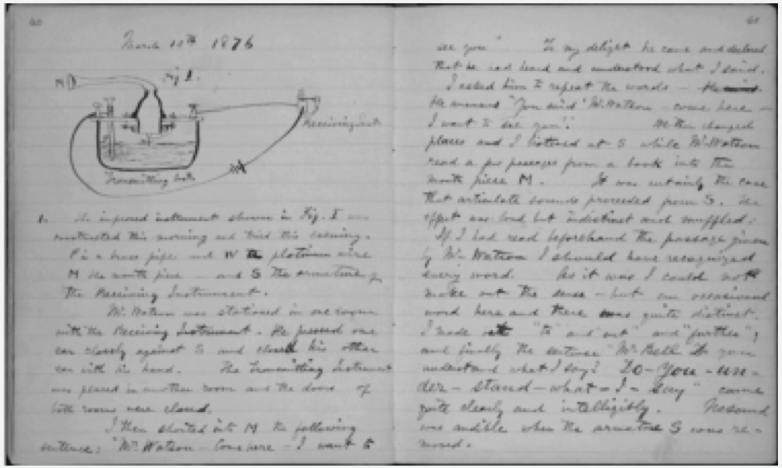
16th August 2018

Problem

Optical Character Recognition


Mom, I just wanted to tell you that Mother's Day wouldn't be possible without me. I'll be waiting for my present in the living room. love, Joshua


PATIENT Jeffrey Daniel LANGLEY D.O.B.
ADDRESS DATE 4/19/13
Rx
Please forgive Mr LANGLEY's tickets due to his unfortunate experience of having a heart attack records prior to having his car accident.
DO NOT REFILL
REFILL TIMES
Use separate form for each controlled substance prescribed
REEN Asbury Park Press


June 10th 1876
Fig 1.
Receiving Instrument
Transmitting Instrument
1. The improved instrument shown in Fig. 1 was constructed this morning and tried this evening. P is a brass pipe, and W the platinum wire. M the mouth piece — and S the armature of the Receiving Instrument.
Mr Watson was stationed in one room with the Receiving Instrument. He pressed one ear closely against S and closed his other ear with the hand. The transmitting instrument was placed in another room and the doors of both rooms were closed.
I then shouted into M the following sentence: "Mr Watson — Come here — I want to see you." To my delight he came and declared that he had heard and understood what I said.
I asked him to repeat the words — He answered "You said 'Mr Watson — come here — I want to see you'." Mr Watson changed places and I listened at S while Mr Watson read a few passages from a book into the mouth piece M. It was nothing like clear that articulate sounds proceeded from S. The effect was loud but indistinct and muffled: If I had read beforehand the passage given by Mr Watson I should have recognized every word. As it was I could not make out the sense — but an occasional word here and there was quite distinct. I made out "to" and "out" and "further" and finally the sentence "Mr Bell do you understand what I say? Yo-You — un-der-stand-what-I-say" came quite clearly and intelligibly. Profound was audible when the armature S was removed.

Dataset

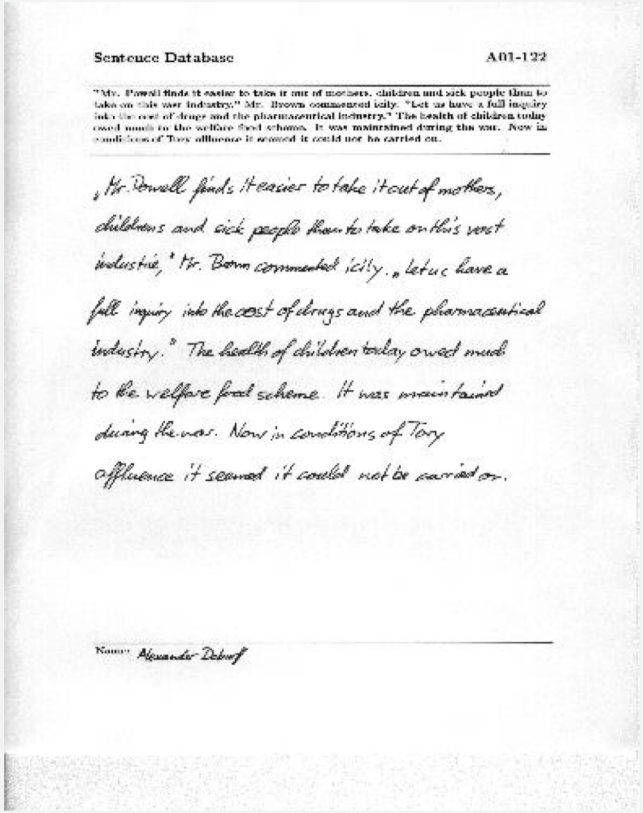
IAM Handwriting Database

Dataset

IAM Handwriting Database:

657 writers
1,539 pages of scanned text
5,685 isolated and labeled sentences
13,353 isolated and labeled text lines
115,320 isolated and labeled words

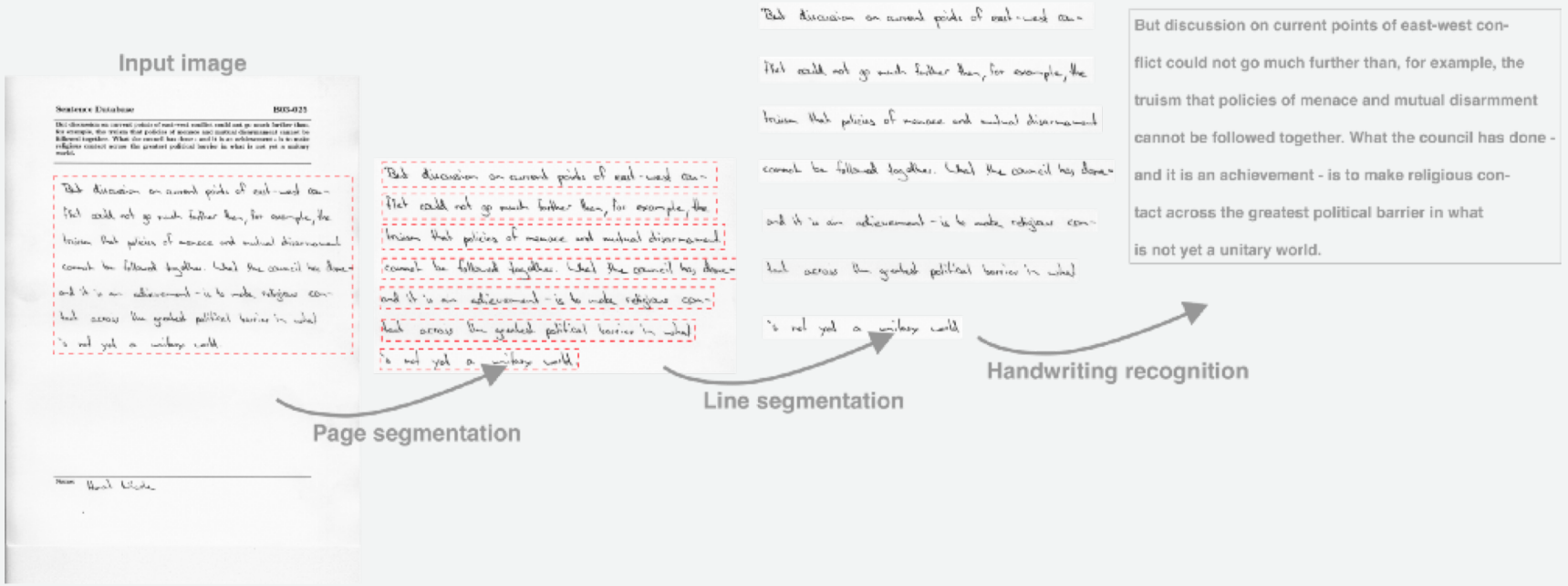
http://www.fki.inf.unibe.ch/databases/iam-handwriting-database
(Register for username and password)



Solution

Handwriting Recognition Pipeline

Pipeline


Input image
Page segmentation
Line segmentation
Handwriting recognition
But discussion on current points of east-west conflict could not go much further than, for example, the truism that policies of menace and mutual disarmment cannot be followed together. What the council has done - and it is an achievement - is to make religious contact across the greatest political barrier in what is not yet a unitary world.

1) Page Segmentation

Page Segmentation

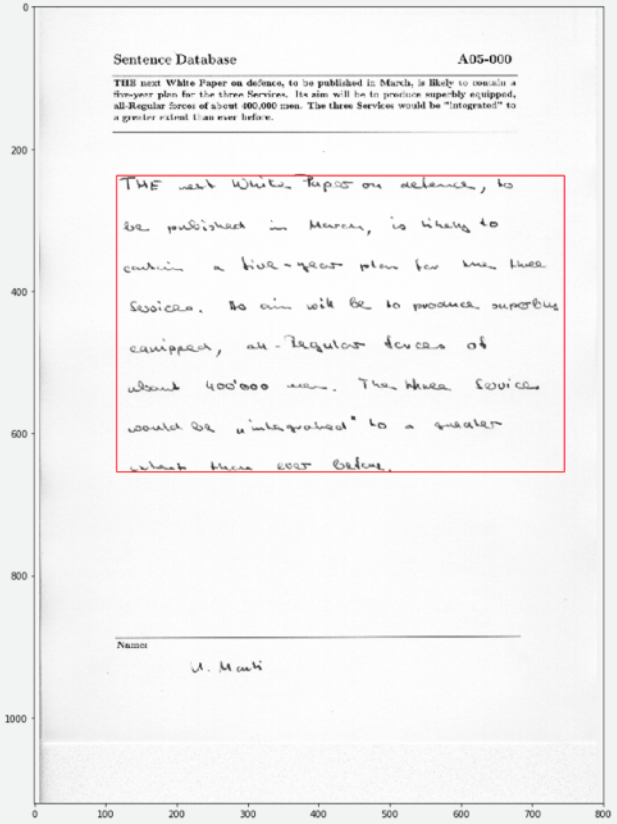
Input: Image with handwritten text region

Output: Bounding box of handwritten text

Problem: Single object detection

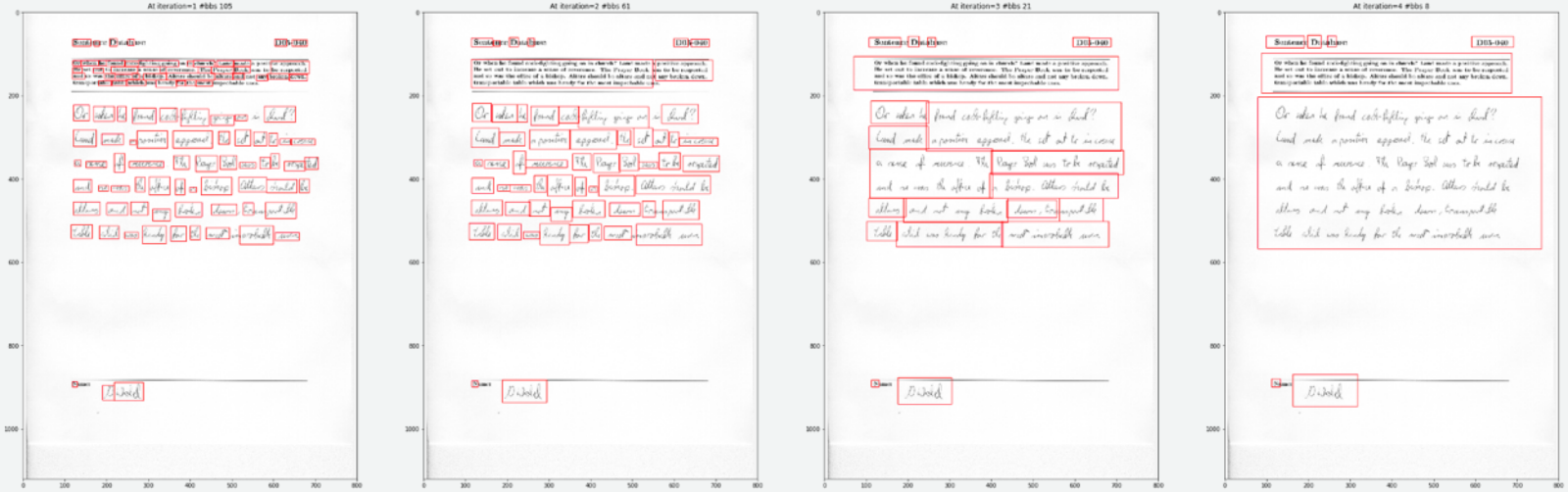
Solutions:

1) Using MSERs
2) Using CNNs

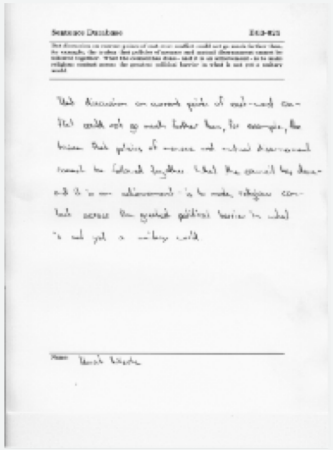


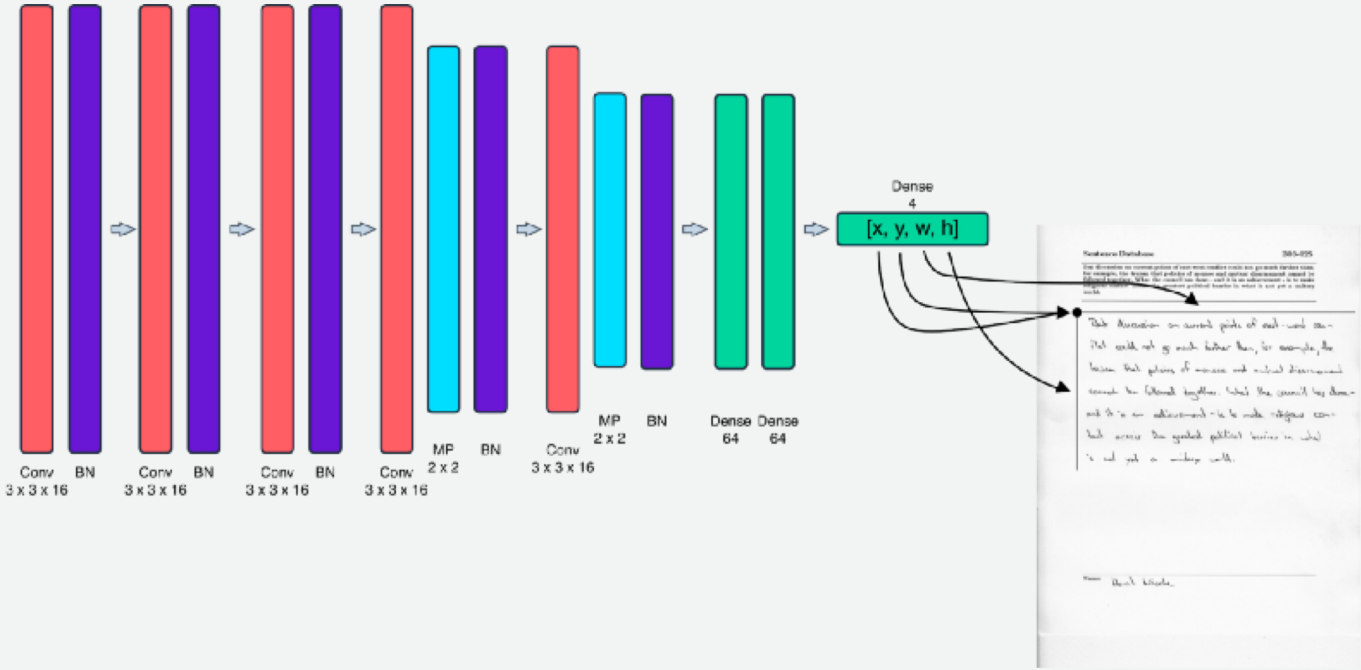
Using MSERs

Maximally Stable Extremal Regions (MSERs) algorithm

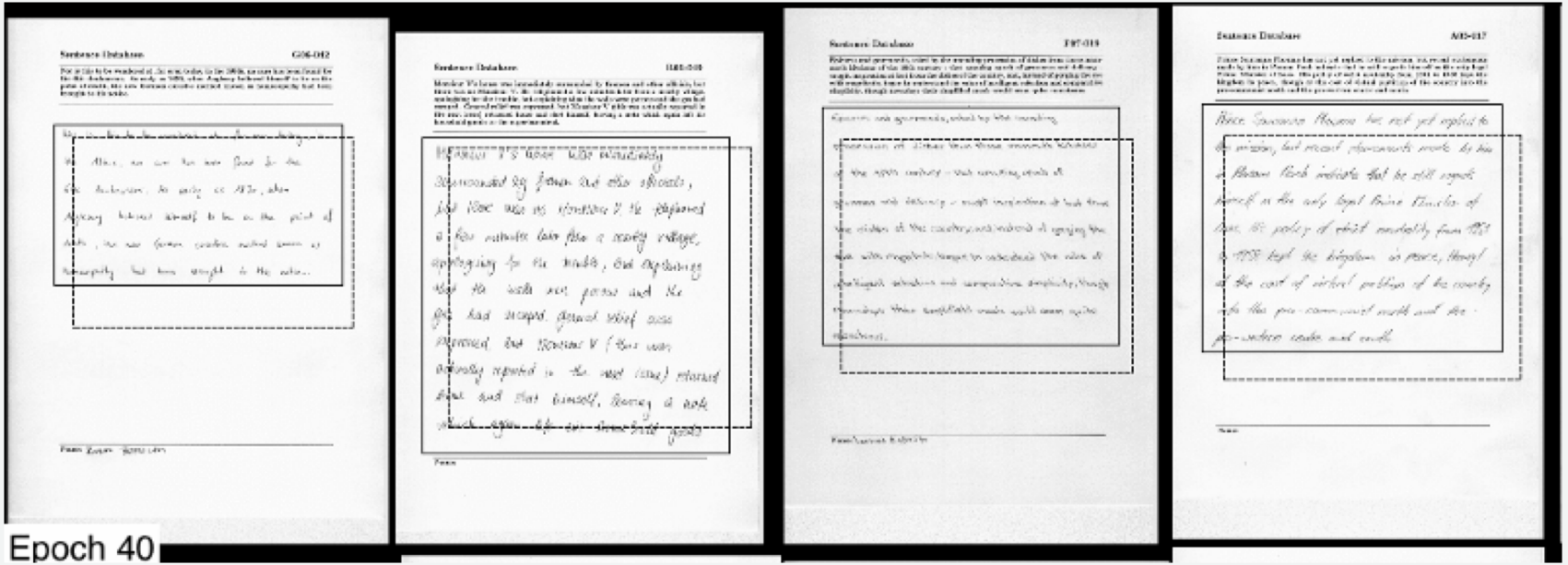


Using CNN




Conv 3 x 3 x 16
BN
Conv 3 x 3 x 16
BN
Conv 3 x 3 x 16
BN
Conv 3 x 3 x 16
MP 2 x 2
BN
Conv 3 x 3 x 16
MP 2 x 2
BN
Dense 64
Dense 64
Dense 4
[x, y, w, h]

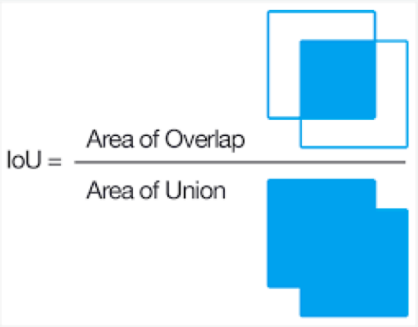
CNN: Data Augmentation

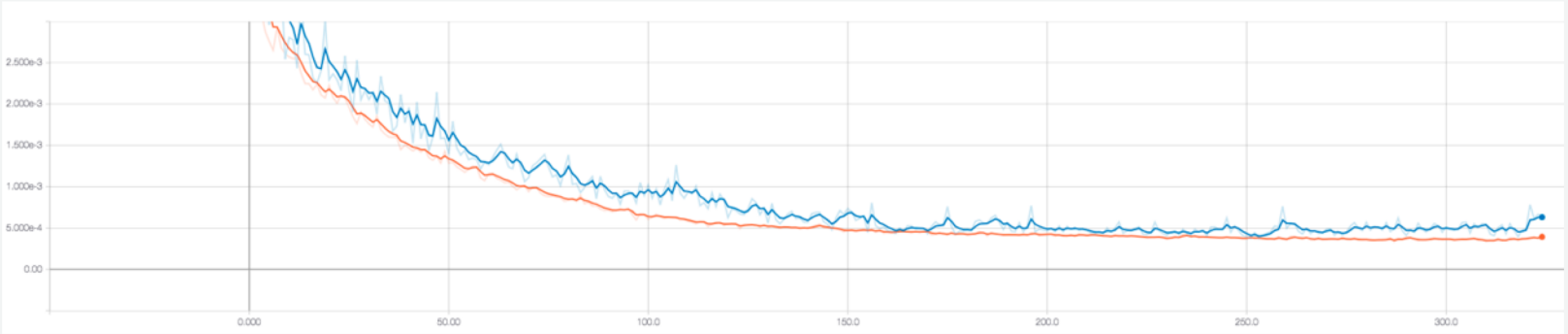


Epoch 40

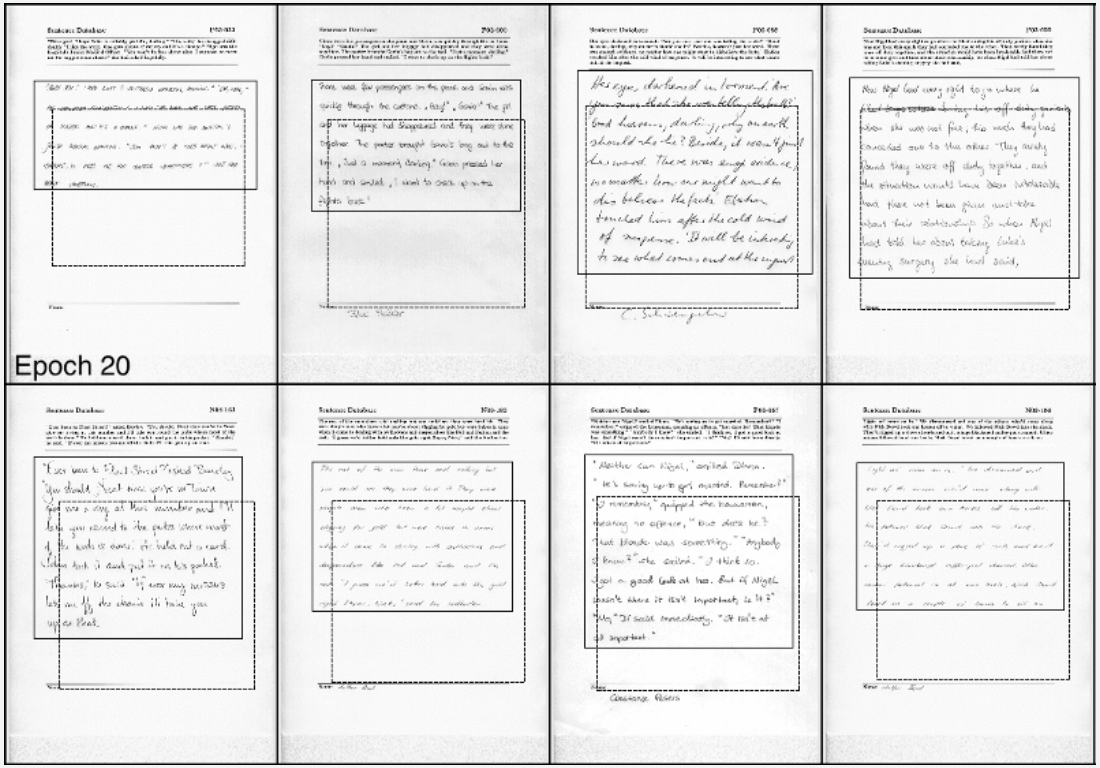
CNN: Training

Start by optimizing Mean Square Error.
When consistent overlap, then fine-tune IOU.





CNN: Results



2) Line Segmentation

Line Segmentation

Input: Image containing only handwritten text
Output: Bounding boxes for each handwritten line

Problem: Multiple object detection

Solutions:

1) Using SSD for lines
2) Using SSD for words

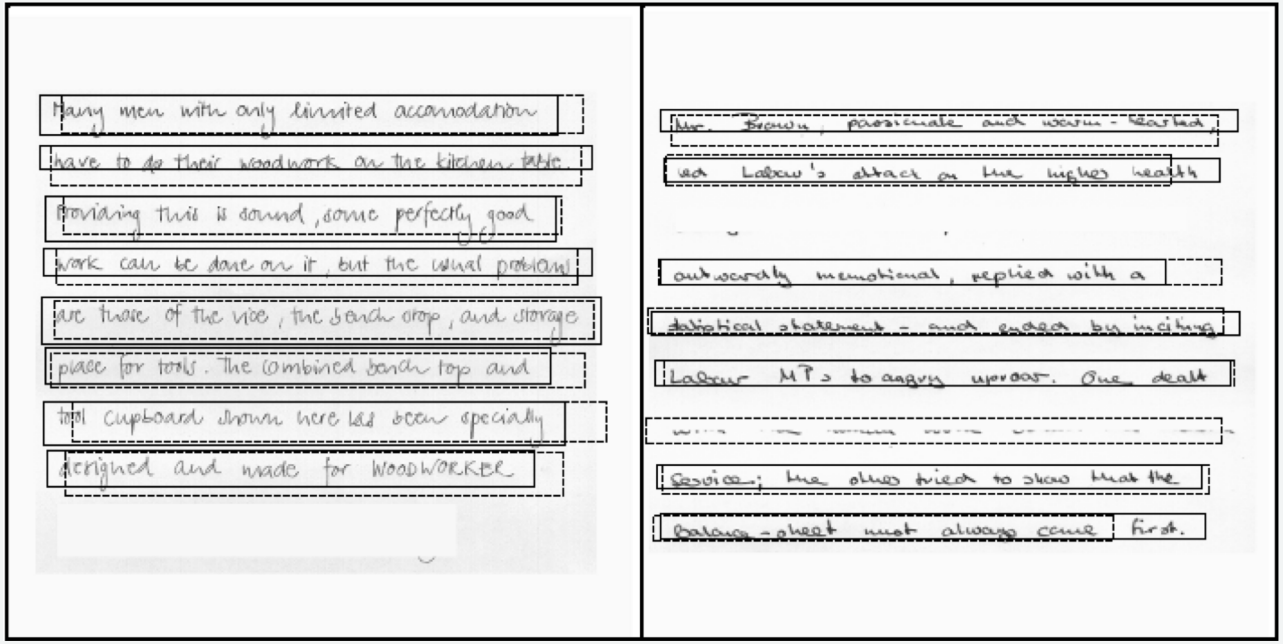
Using SSD

SSD: Single Shot Multi-box Detector

Customized:

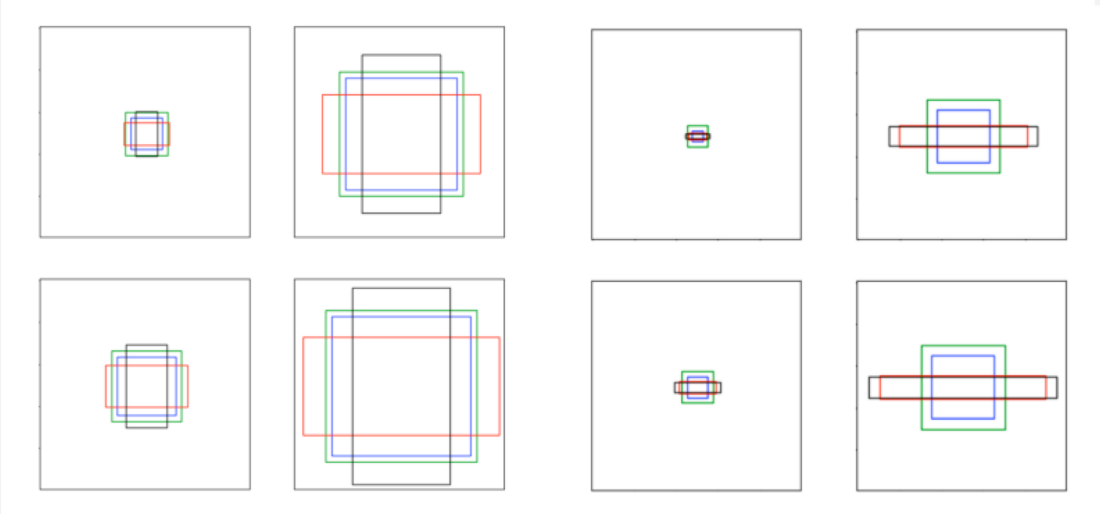
1. Data augmentation
2. Anchor boxes
3. Network architecture
4. Non-maximum suppression

Line SSD: Data Augmentation



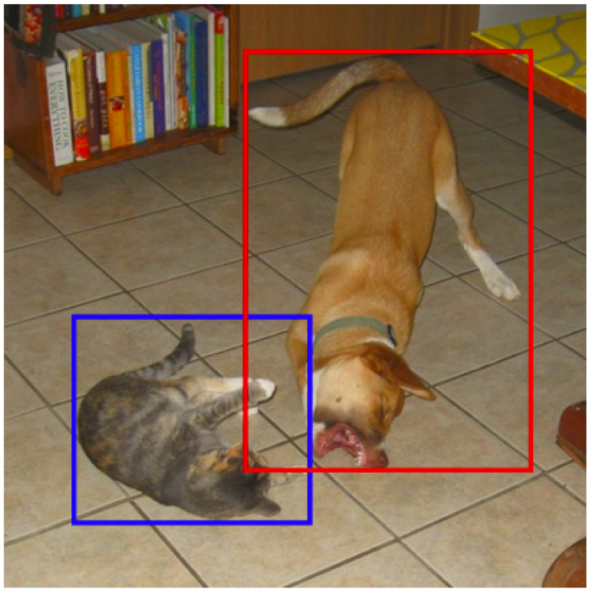
Line SSD: Anchor boxes

Standard *Custom*

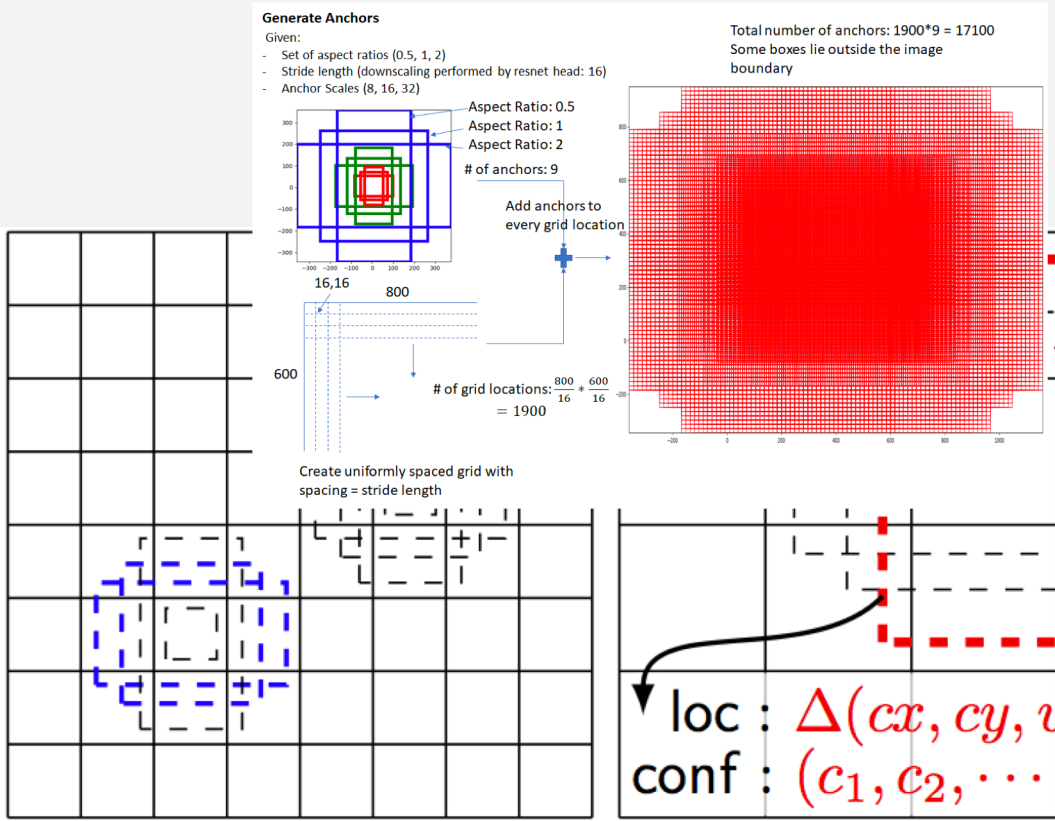


`mxnet.ndarray.contrib.MultiBoxPrior`

Line SSD: Anchor boxes



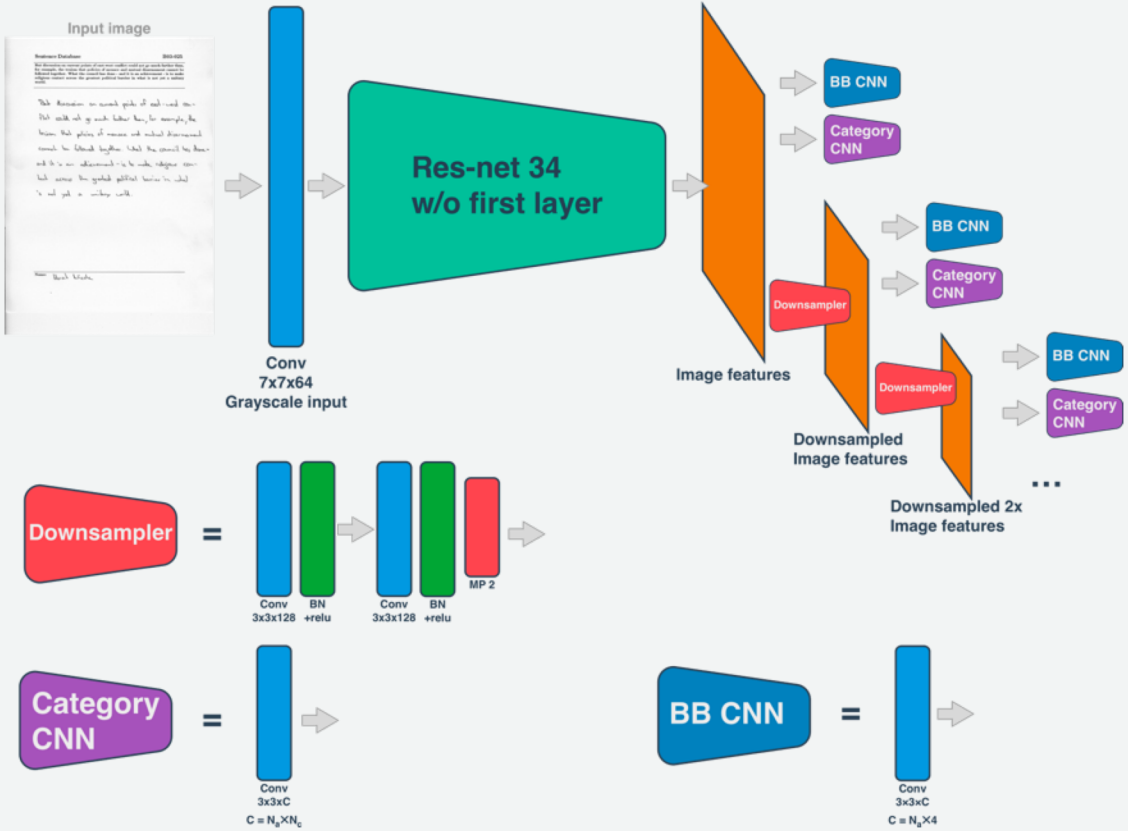
(a) Image with GT boxes



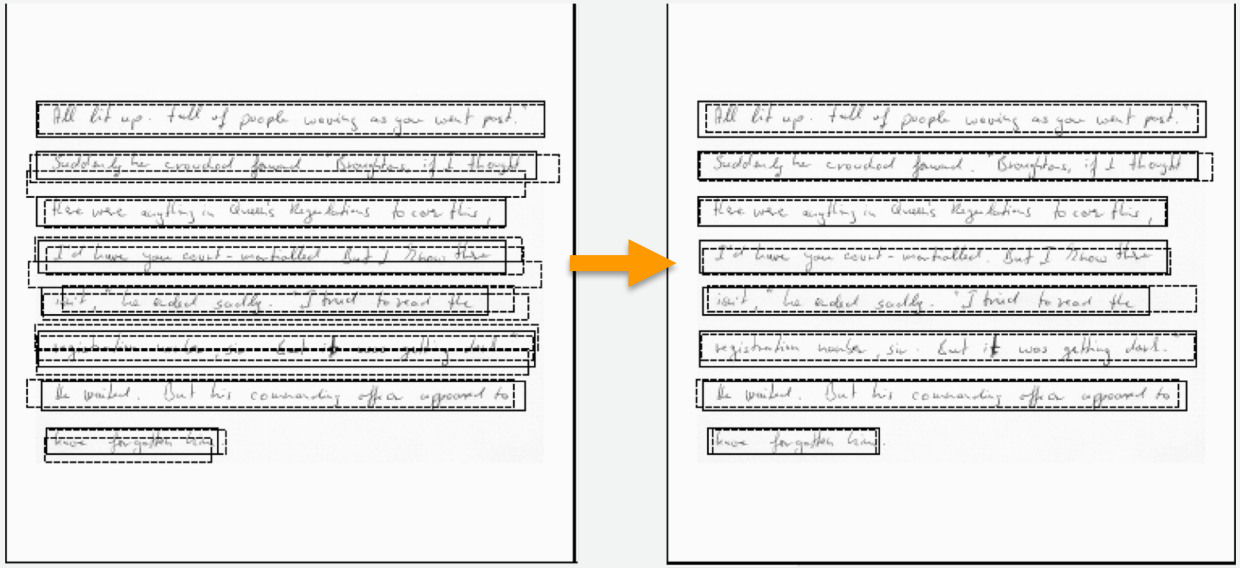
(b) 8×8 feature map

(c) 4×4 feature map

Line SSD: Network architecture



Line SSD: Non-Maximum Suppression

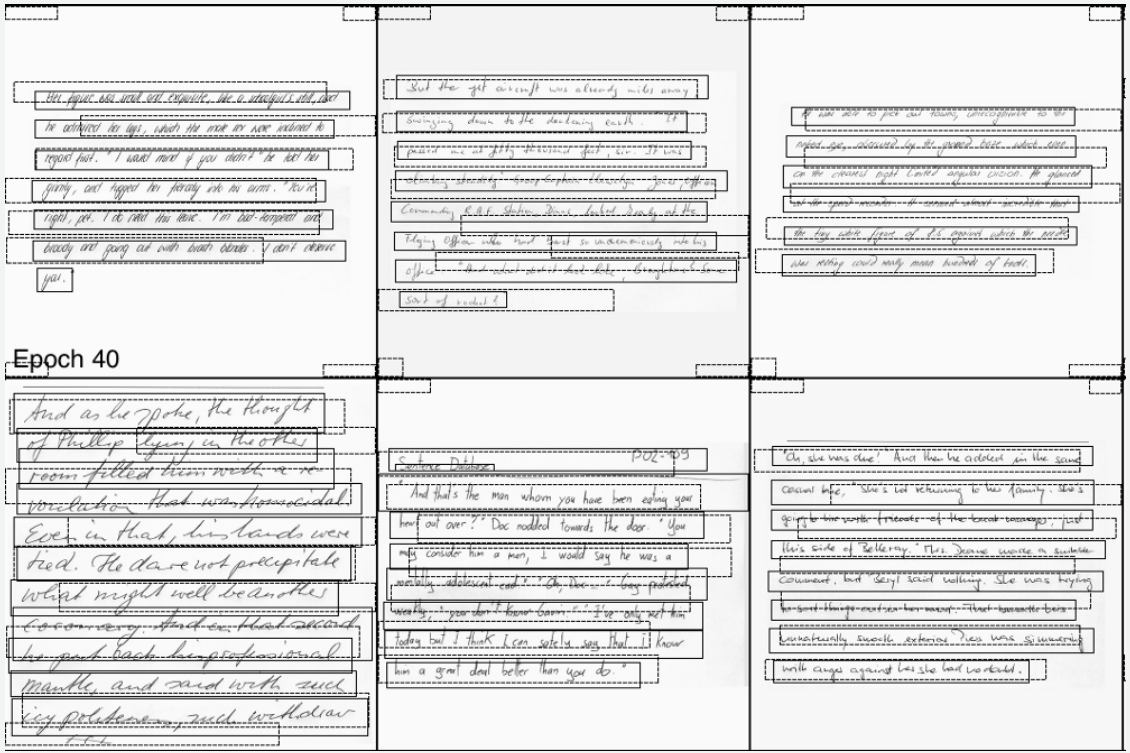


a) without non-maximum suppression

b) with non-maximum suppression

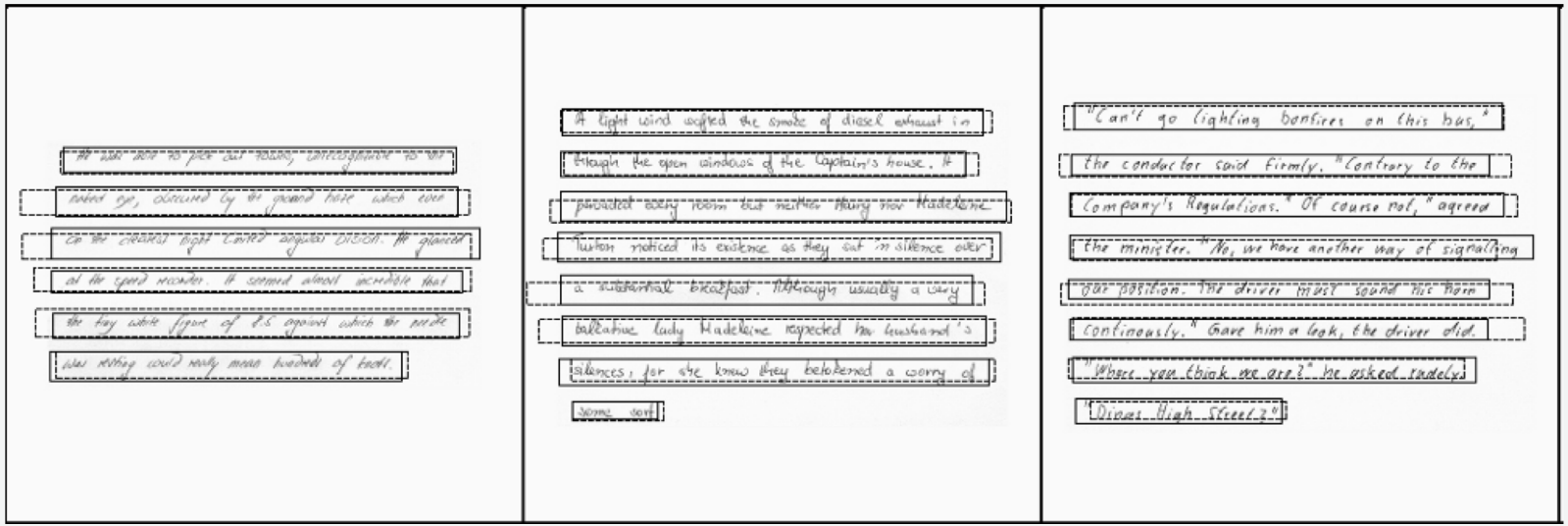
`mxnet.ndarray.contrib.box_nms`

Line SSD: Training

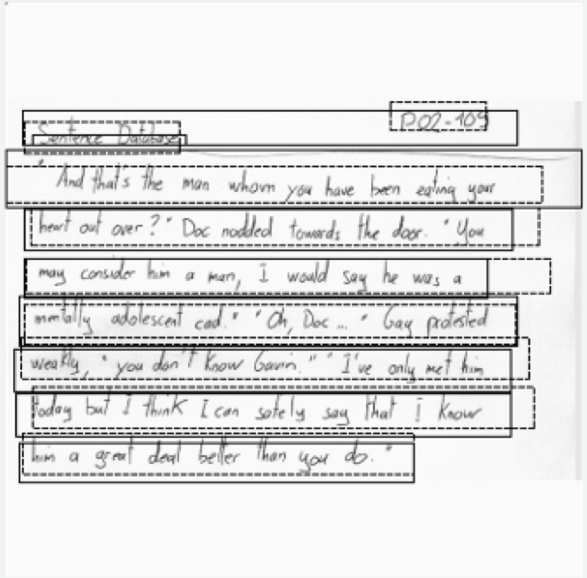


Line SSD: Results

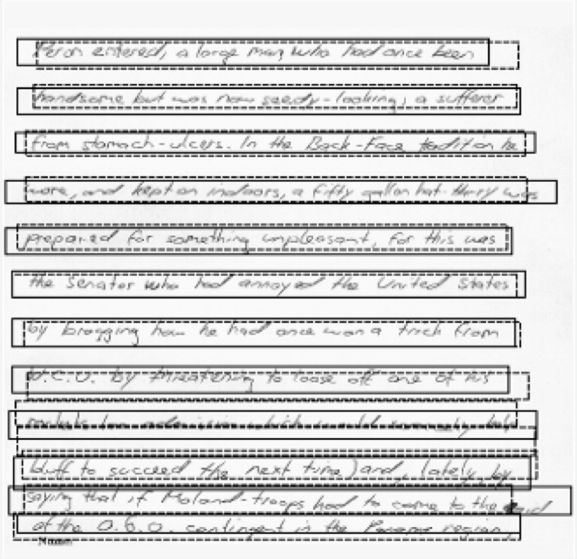
Train IOU = 0.593 & Test IOU = 0.573



Line SSD: Results



b) Ignored incorrectly labelled smudge
(second line from the top)



c) Failure cases: misalligned predicted box
(forth line from the bottom)

Word SSD

3) Handwriting Recognition

Handwriting Recognition

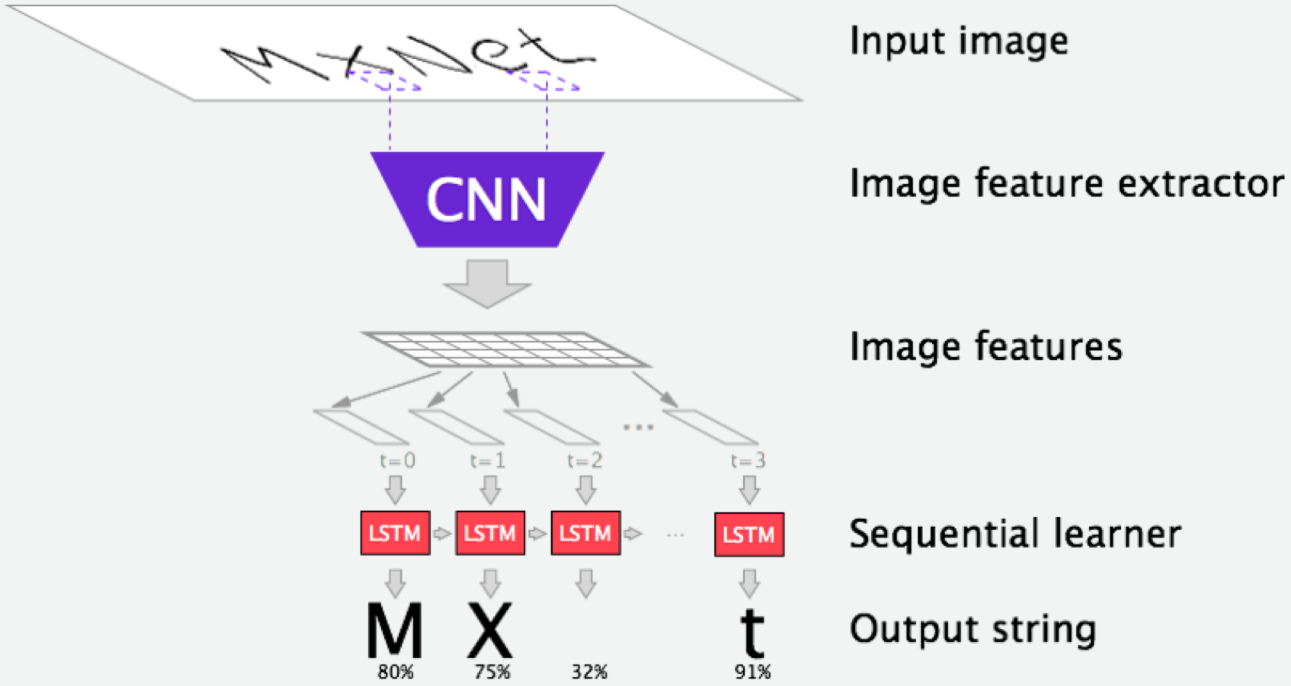
Input: Image of single line of handwritten text

Output: Probabilities of each character for each time step.

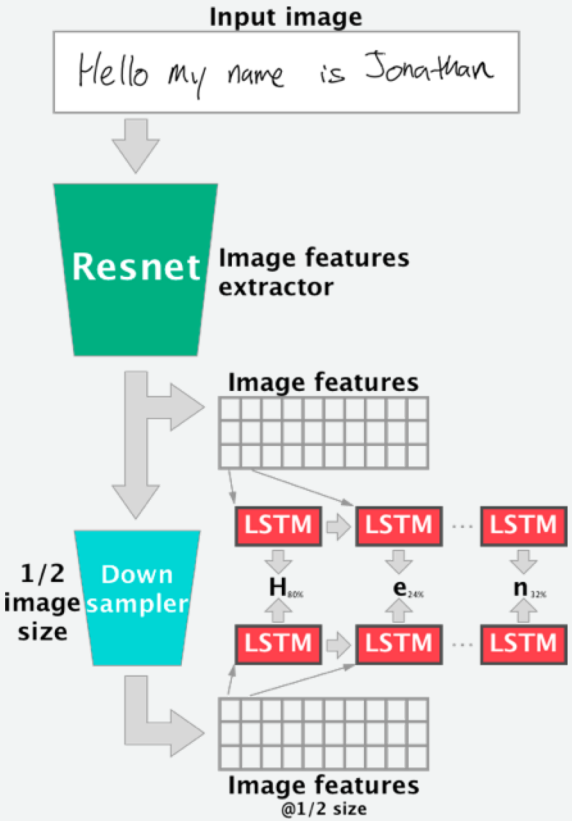
i.e. an array of shape (sequence_length × vocab. of characters)

Inference Output: String of recognized text

Detection: Using CNN-BiLSTM



Detection: Adding down-sampling



Detection: With CTC loss

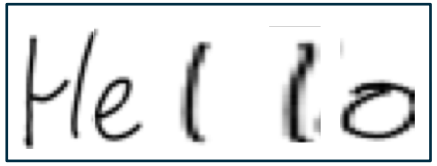
Problem:

- Must have label for each time step.
- Or must model alignment.
- Must compress final sequence.

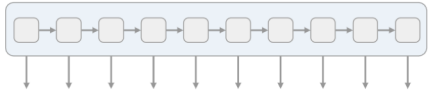
Solution:

- Connectionist Temporal Classification loss.
- Defines a collapsing function.
- Assumes monotonic and many-to-one.

`mxnet.ndarray.contrib.CTCLoss`



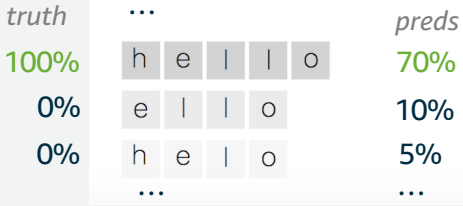
We start with an input sequence,



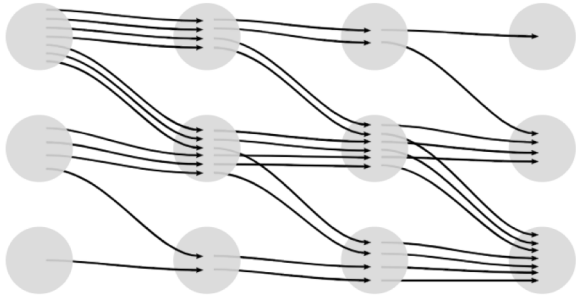
The input is fed into an RNN, for example.

The network gives $p_t(a \mid X)$, a distribution over the outputs {h, e, l, o, ϵ} for each input step.

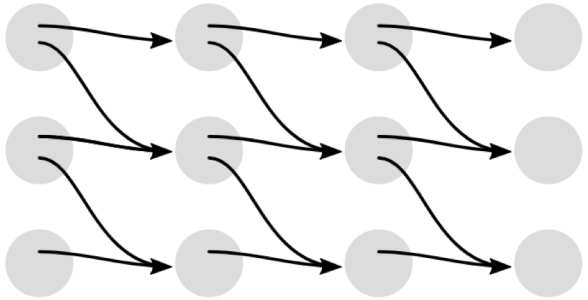
With the per time-step output distribution, we compute the probability of different sequences



By marginalizing over alignments, we get a distribution over outputs.

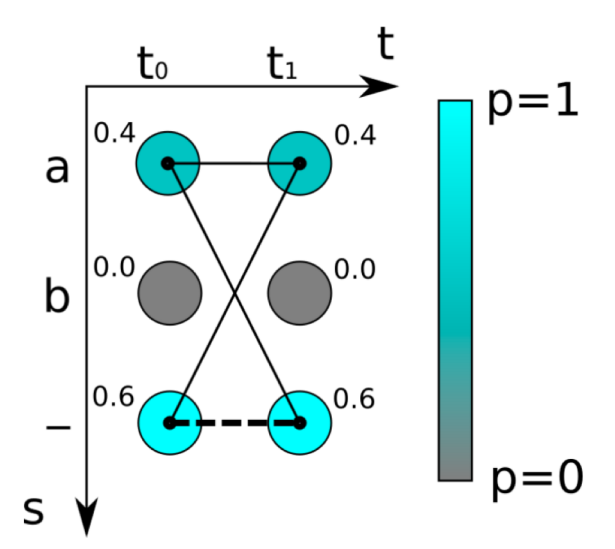


Summing over all alignments can be very expensive.



Dynamic programming merges alignments, so it's much faster.

Inference: Best Path vs Correct decoding



$$Best\ Path = (-,-)$$
$$Best\ Path\ Output = C(-,-) = \text{“”}$$

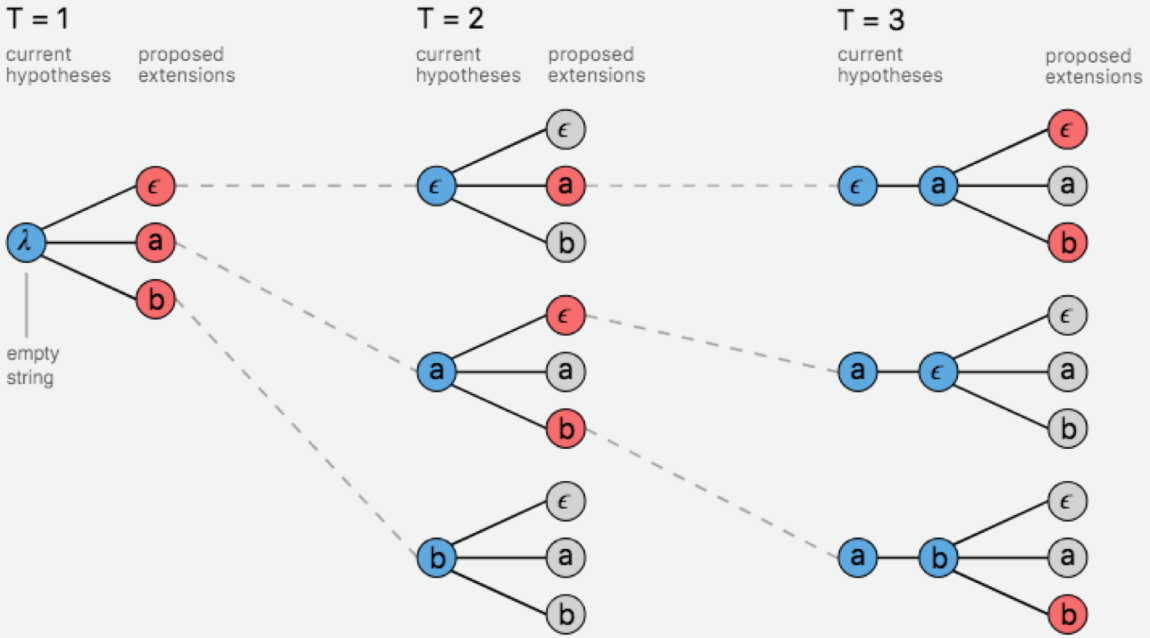
$$P(\text{“”}) = P(-,-) = 0.36$$
$$P(\text{“}a\text{”}) = P(\text{“}-a\text{”}) + P(\text{“}a-\text{”}) + P(\text{“}aa\text{”})$$
$$= 0.6 * 0.4 + 0.4 * 0.6 + 0.4 * 0.4$$
$$= 0.64$$
$$Correct\ Output = \text{“}a\text{”}$$

Can’t check all paths though.

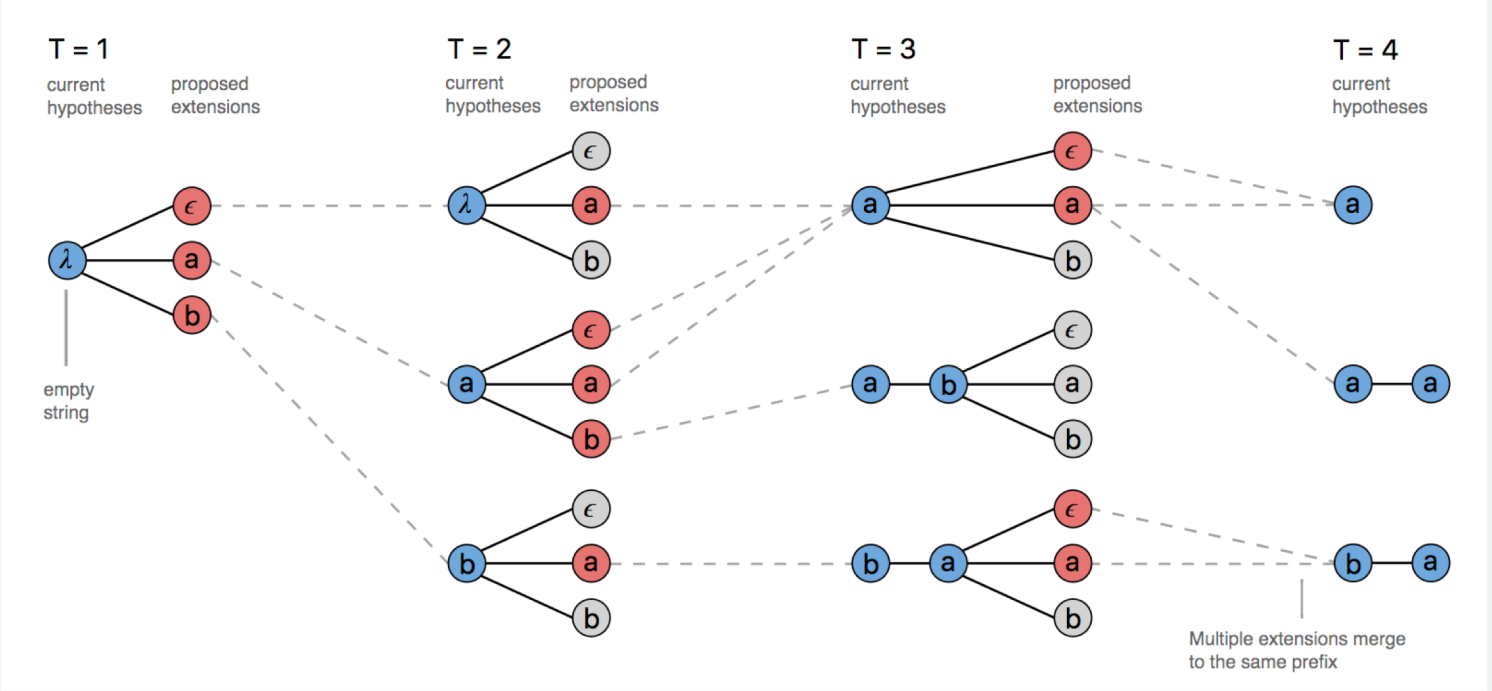
$alphabet\_length^{sequence\_length}$ paths.

i.e. 81^{30} paths!

Inference: With Beam Search



Inference: With CTC Beam Search



Inference: Adding Lexicon Search

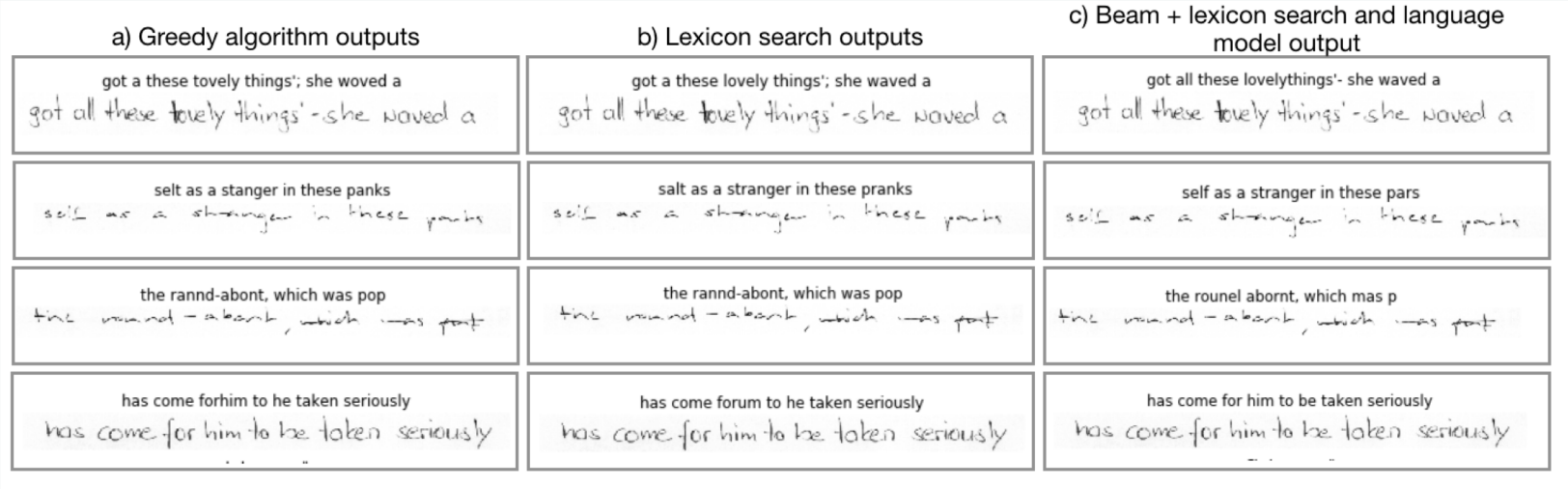
```
arg_max_string = get_arg_max_output(network_output)
cont = Contractions('word2vec model')
decontracted_text = list(cont.expand_texts([arg_max_string])[0])
words = tokenize(decontracted_text)
output_words = []
for word in words:
    suggested_words = suggest_words(word)
    dists = []
    for suggested_word in suggested_words:
        dists.append(edit_distance(suggested_word, word))
    output_words.append(suggested_words[arg_min(dists)])
output_string = detokenize(output_words)
contracted_text = list(cont.contract_texts([output_string])[0])
```

Inference: Adding Language Model

```
# using pretained language model

lm_model, vocab = gluonnlp.model.get_model('awd_lstm_lm_1150', pretrained=True)
hidden = lm_model.begin_state(batch_size=1, func=mx.nd.zeros)

tokens = tokenize(line_string)
token_ids = [vocab[token] for token in tokens]
data = mx.nd.array(token_ids).expand_dims(1)

output, hidden = lm_model(data, hidden)
```

And evaluate ‘perplexity’ of of each candidate and take the one with lowest value as final prediction.

Inference: Results



Mean Character Error Rate (CER)

Greedy = 21.170.

Greedy + Lexicon Search: 21.152.

Beam Search + Lexicon Search: CER = 21.058.

Thanks!

And don’t forget to check out:
https://medium.com/apache-mxnet
https://github.com/ThomasDelteil/Gluon_OCR_LSTM_CTC