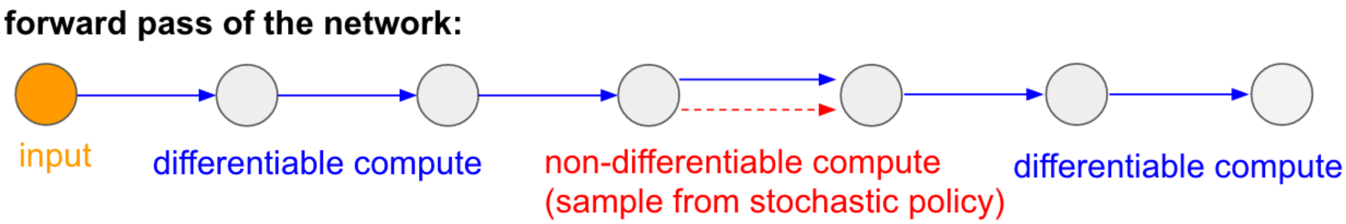
Differentiable programming with bypass neural nets

Nilanjan Ray
Associate Professor
Department of Computing Science
University of Alberta

nray1@ualberta.ca
https://webdocs.cs.ualberta.ca/~nray1/

Heterogeneous Functional Compositions (HFC)



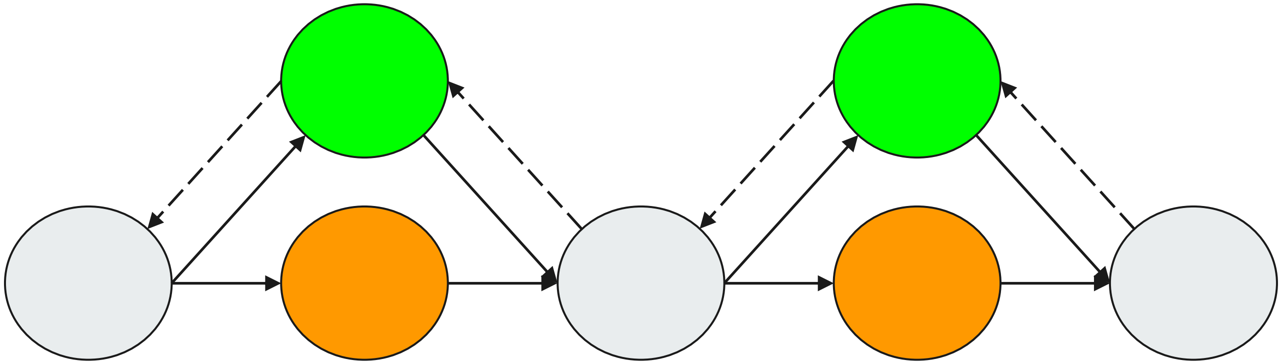
Mixing **differentiable** and **non-differentiable** computations

From Dr. Andrej Karpathy's blog: http://karpathy.github.io/2016/05/31/rl/

I am referring to this type of processing pipeline as HFC

Using bypass neural nets in HFC

Differentiable module | Bypass neural net
Non-differentiable module | Forward pass
Backward pass



Differentiable bypasses in Heterogeneous Functional Composition

Bypass net: A closer look

$F(I;\theta)$: Non-differentiable functional module with input *I* and parameter θ

$N(I,\theta;\varphi)$: Bypass neural network with input (I,θ) and its own parameter φ

Bypass net approximates the output of the non-differentiable functional module:

$$min_{\varphi}||N(I,\theta;\varphi) - F(I;\theta)||$$

Theory (Hornik et al.) shows that bypass network will also approximate a generalized gradient of the non-differentiable functional module

Hornik, K., Stinchcombe, M.B., White, H., 1990. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. Neural Networks 3, 551–560.

Learning bypass net with perturbation

For correct gradient approximation perturbation of the inputs is required:

$$min_\varphi \sum_{\substack{i=1 \\ \epsilon_i^1 \sim \mathcal{N}(0,\sigma_1) \\ \epsilon_i^1 \sim \mathcal{N}(0,\sigma_2)}}^{n} \|N(I+\epsilon_i^1, \theta+\epsilon_i^2; \varphi) - F(I+\epsilon_i^1; \theta+\epsilon_i^2)\|$$

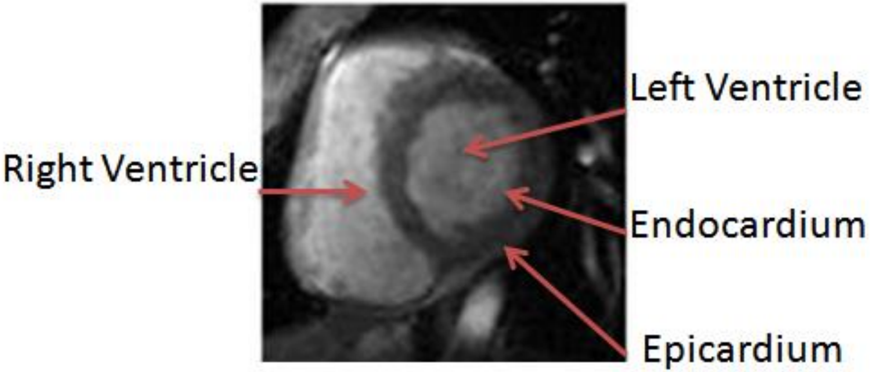
Learning in HFC

```
for data batch in the training set do
    A. Compute forward pass by pushing data batch through all
       functional modules;
    for each bypass neural net do
        for i in n do
            1. Sample noise from a Gaussian distribution;
            2. Perturb input and parameters of the non-differentiable
               functional module by noise;
            3. Compute output from functional module;
            4. Take a forward and a backward pass to adjust parameters
               of the bypass network;
        end
    end
    B. Compute backward pass through all differentiable functional
       modules and all bypass networks;
end
```

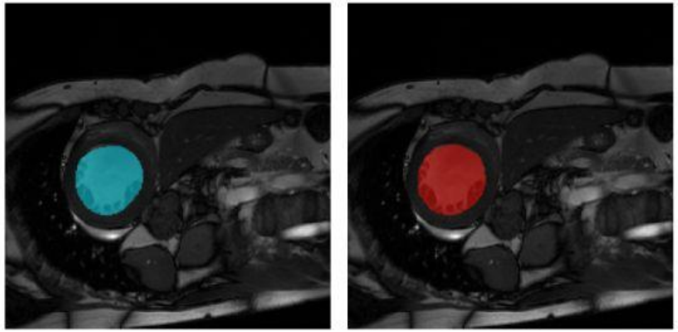
Algorithm 1: Learning Heterogeneous Functional Composition

Applications

Application 1: Left ventricle segmentation

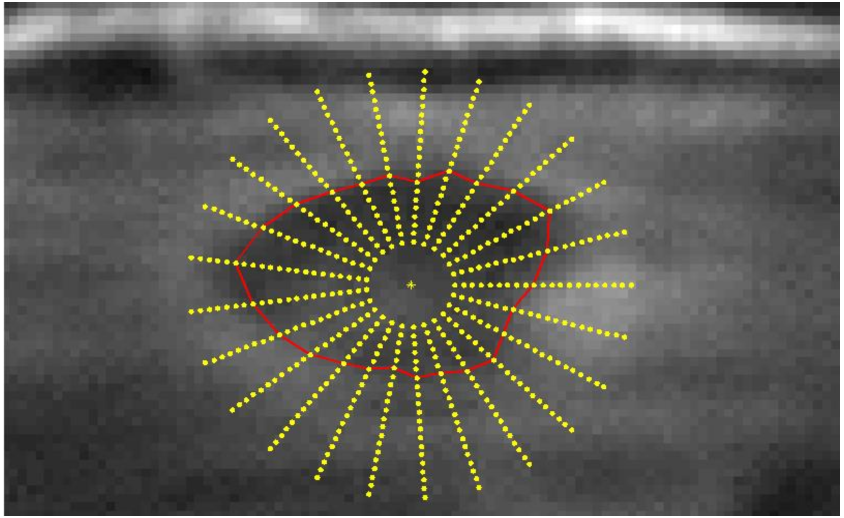


Example LV segmentations



http://article.sapub.org/10.5923.j.ajbe.20120203.07.html

Traditional tool: Dynamic programming for blob object segmentation



The goal of dynamic programming is to find exactly one graduation point on a radial line to delineate the object boundary

DP: Discrete optimization setup

Applies when objective/cost function has an additive and overlapping structure:

$$\min_{v_1,\ldots,v_N} E(N, v_N, v_1) + \sum_{n=1}^{N-1} E(n, v_n, v_{n+1})$$

$$E(n,i,j) = \begin{cases} dg(n,i) + dg(n \oplus 1, j), \text{if } |i-j| \le \delta, \\ \infty, \ \text{otherwise}, \end{cases}$$

where $dg(n, i) = g(n, i) - g(n, i+1)$ is the directional derivative of image on radial line n at graduation mark i.

DP: Algorithm

DP algorithm builds two arrays:

Value and index functions as shown in the algorithm.

Then, it backtracks to find a solution.

DP is a type of shortest path (SP) algorithm.

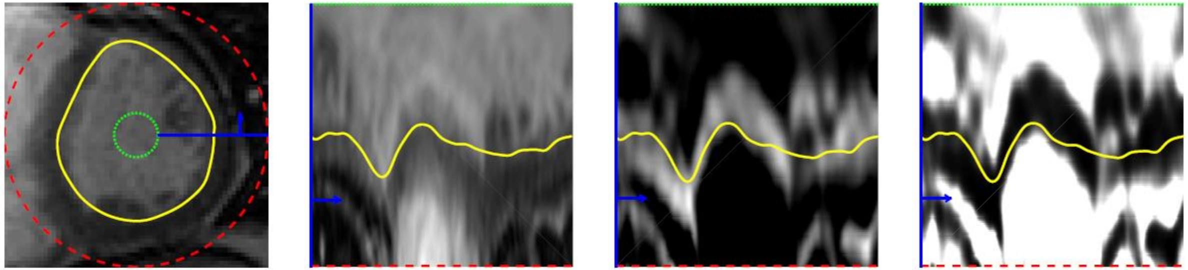
Incidentally, SP is a special type of linear programming.

And, linear programming is a convex and continuous optimization problem!

/* Construct value function U and index function I */
for $n = 1, \dots, N-1$ **do**
  for $i, k = 1, \dots, M$ **do**
    if $n == 1$ **then**
      $U(1, i, k) = \min_{1 \leq j \leq M}[E(1, i, j) + E(2, j, k)]$;
      $I(1, i, k) = \text{argmin}_{1 \leq j \leq M}[E(1, i, j) + E(2, j, k)]$;
    else
      $U(n, i, k) = \min_{1 \leq j \leq M}[U(n-1, i, j) + E(n+1, j, k)]$;
      $I(n, i, k) = \text{argmin}_{1 \leq j \leq M}[U(n-1, i, j) + E(n+1, j, k)]$;
    end
  end
end
/* Backtrack and output $v(1), \dots, v(N)$ */
$v(1) = \text{argmin}_{1 \leq j \leq M}[U(N-1, j, j)]$;
$v(N) = I(N-1, v(1), v(1))$;
for $n = N-1, \dots, 2$ **do**
  $v(n) = I(n-1, v(1), v(n+1))$;
end

Algorithm 1: Dynamic programming

DP: LV segmentation….

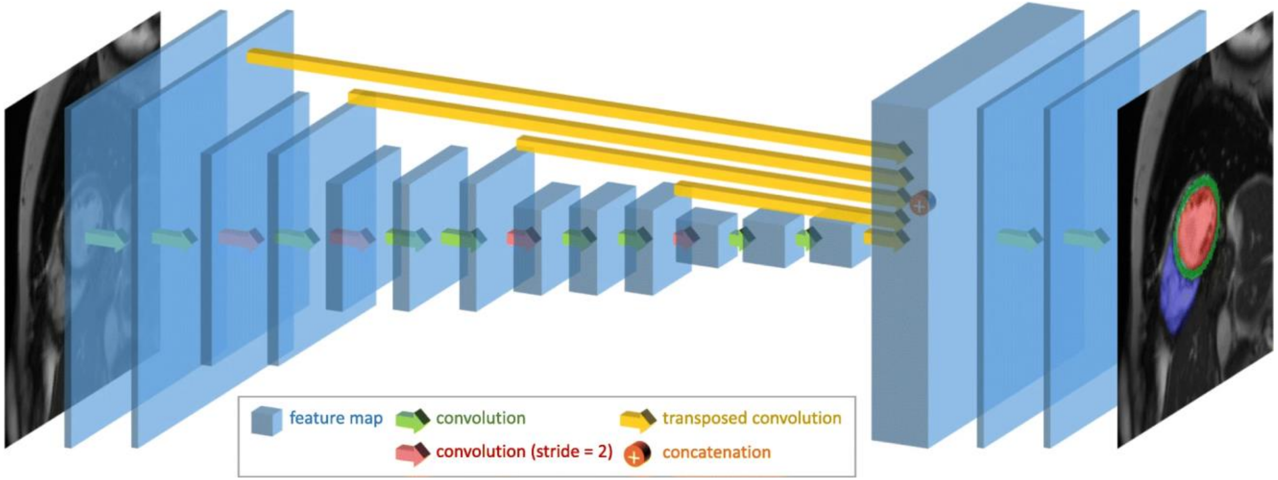


(a) (b) (c) (d)

Fig. 3. (a) Original LV image, $I(x, y)$; (b) image in polar coordinates, $I_{\mathcal{P}}(r, \theta)$; (c) image gradient, $I_{\mathcal{G}}(r, \theta)$; and (d) edge map, $e_{\mathrm{MAP}}(r, \theta)$. The yellow line corresponds to the LV segmentation. The green and red lines correspond to the minimum and maximum radius, respectively, and the blue line and arrow help illustrate the conversion to polar coordinates.

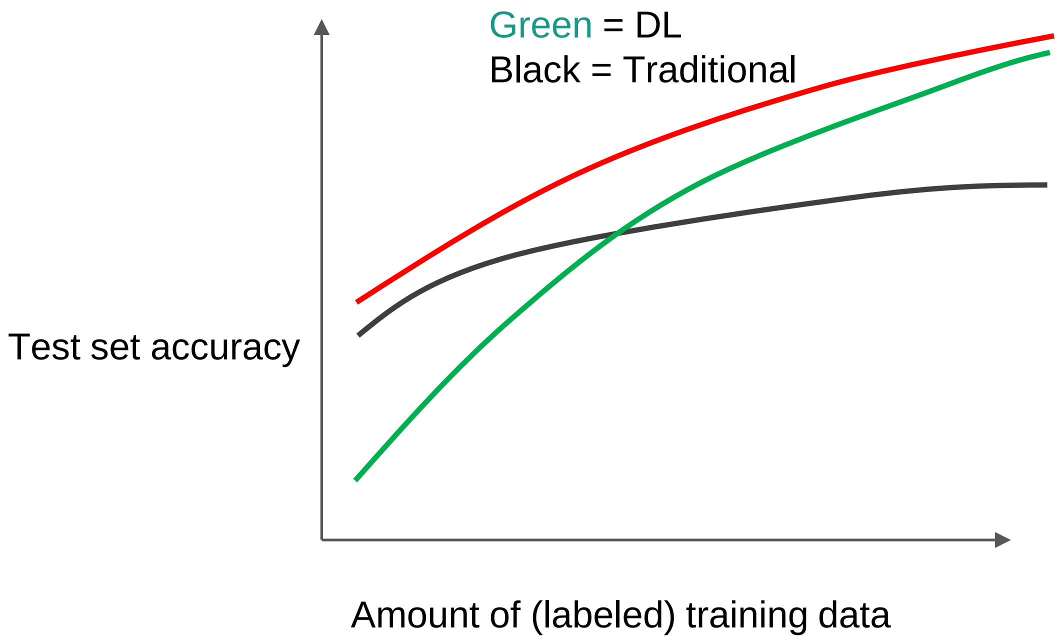
https://www.researchgate.net/publication/320859355

Fully convolutional net-based segmentation



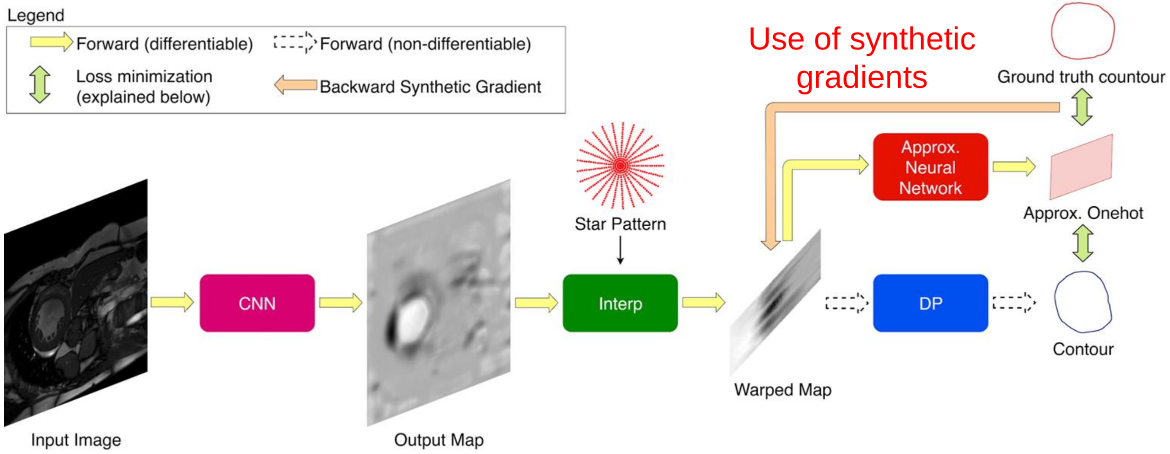
Bai et al.: https://jcmr-online.biomedcentral.com/articles/10.1186/s12968-018-0471-x

CNN vs. conventional algorithm



How do we achieve the red curve?

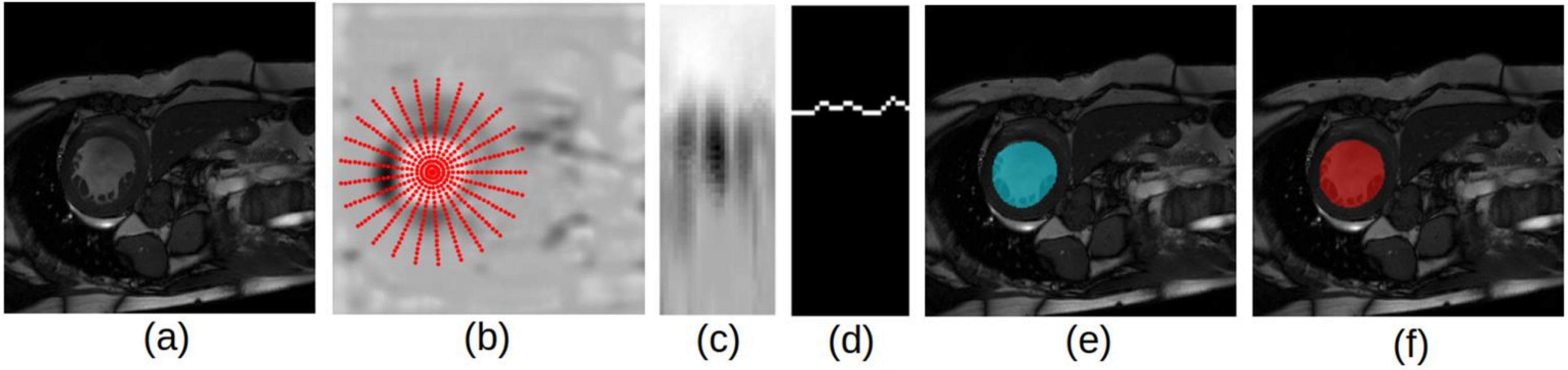
End-to-end Convnet + Dynamic Programming



EDPCNN: End-to-end learning with CNN + Dynamic Programming

Nguyen and Ray, https://2020.midl.io/papers/nguyen20a.html

EDPCNN processing pipeline



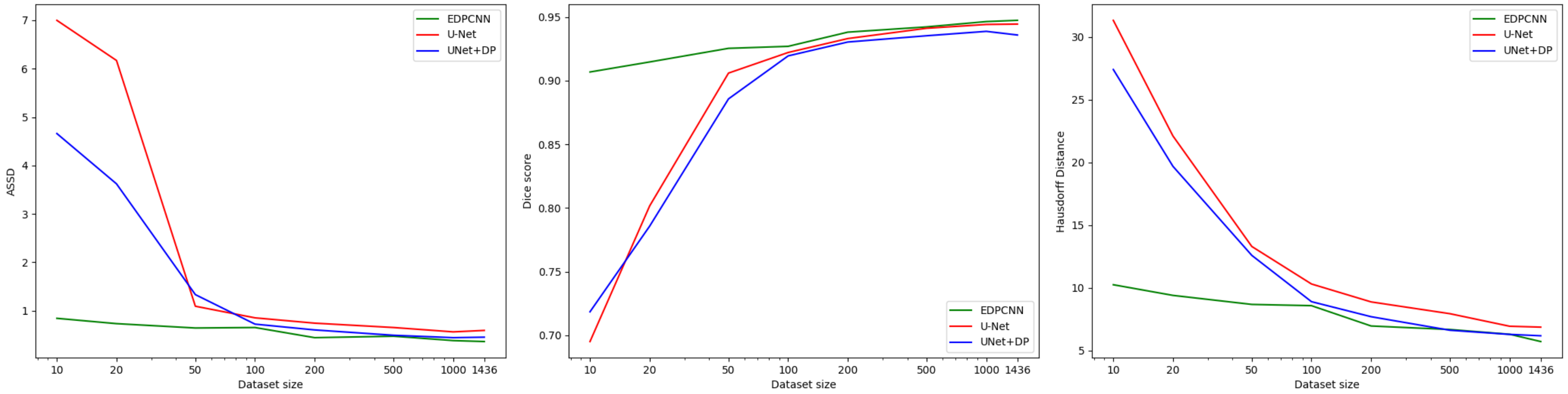
Illustrations of processing pipeline: (a) input image, (b) Output Map with an example star pattern, (c)Warped Map and (d) output indices indicating LV on the warped space (e) segmentation obtained with EDPCNN (f) ground truth.

End-to-end training by synthetic gradients

for $I, p_{gt} \in Training\ \{Image, Ground\ \ Truth\}\ \ Batch$ **do**
$g = Interp(Unet(I))$;
Initialize s to 0;
while $s < S$ **do**
Sample ε_s from $\mathcal{N}(0; I)$;
$\min_\phi L(F(g + \sigma\varepsilon_s), DP(g + \sigma\varepsilon_s))$;
$s = s + 1$;
end
$\min_\psi \ L(F(g), p_{gt})$;
end

Perturbation is crucial!

Results: End-to-end training is sample efficient



Results on ACDC dataset: https://2020.midl.io/papers/nguyen20a.html

EDPCNN: Computation time

Table 1. Computation time on an NVIDIA GTX 1080 TI

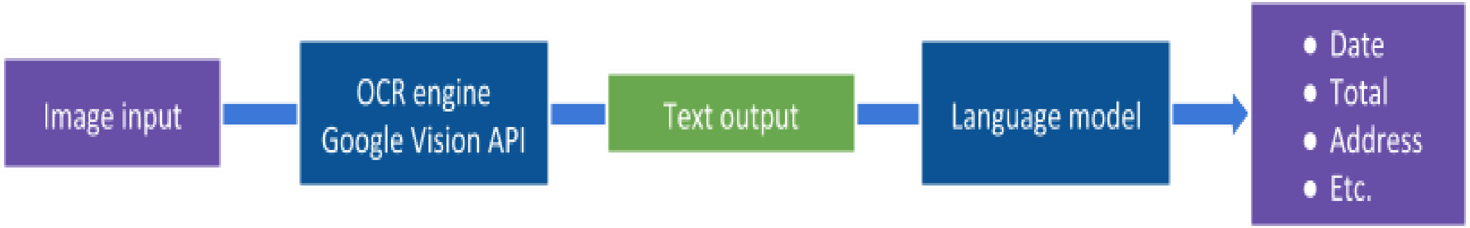
Method	Time / Training iteration	Total iterations	Total training time	Inference time / Image
U-Net	0.96s	20000	5h 20m	0.01465s
EDPCNN	1.575s	20000	8h 45m	0.01701s

64% increase in training time

16% increase in test time

Application 2: Image Pre-processing for OCR

Accuracy of OCR depends on the quality of input document image



Joint work with Ayantha Randika

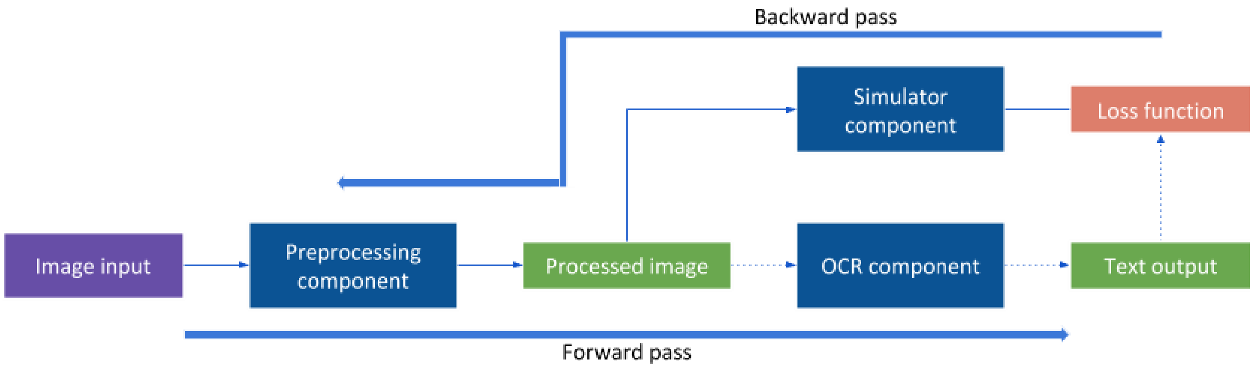
Proposed Solution: OCR-aware preprocessing

- Blindly using conventional image processing techniques to improve images may not improve commercial OCR engines.

- We need a method to do pixel level manipulations to improve the shape of characters in addition to colour.

OCR-aware image enhancement

- However, training a such preprocessor for a black-box problematic since there is no direct way to get error feedback at the pixel level.
- Therefore we suggest employing an approximating technique to synthesise the error of the OCR component.



Datasets

We have mainly used two publicly available datasets. Collectively they contain around 2200 receipt documents.

- ICDAR Dataset: Released as part of “ICDAR 2019 Robust Reading Challenge on Scanned Receipts OCR and Information Extraction”
- Find-It Dataset: Released as part of “Find it!” fraud detection contest

ICDAR Dataset

Find-It Dataset

city
CRF-CITY LA ROCHELLE
33 RUE DE LA SCIERIE
17000 LA ROCHELLE
Tel : 05.46.27.02.12

DESCRIPTION	QTE		MONTANT
*1/2 BAGUETTE 125G			0.46€
	1.0222 x	0.45€	
*160G BLC PLT 4TR.F			2.15€
*1L BOISSON MOJITO			2.60€
*320G SALADE ANTIBE			3.74€
*EMMENTAL EN TRANCH			1.91€
*M.NOVA SNACK CHOCO			0.92€

6 ARTICLE(S) TOTAL A PAYER 11.78€

CB EMV SANS CONTACT EUR 11.78€

0001 004 000035 24/02/2017 10:49:08

MERCI DE VOTRE VISITE
A BIENTOT

Carrefour city
CRF-CITY LA ROCHELLE
33 RUE DE LA SCIERIE
17000 LA ROCHELLE
Tel : 05.46.27.02.12

DESCRIPTION	QTE	MONTANT
180G QUINOA BOULG		2.77€
PARMENTIER PATATES		4.14€
VACHE A BOIRE VANI		1.99€

3 ARTICLE(S) TOTAL A PAYER 8.90€

CB EMV SANS CONTACT EUR 8.90€

004 000123 05/09/2016 12:01:19

MERCI DE VOTRE VISITE
A BIENTOT

FRANCESCA
ELIOR CONCESSIONS GARES SAS
11 Boulevard de Vaugirard 75015 Paris
Tél : 01 43 21 51 23
RCS NANTERRE 524595236 TVA FR76524959236

SECOURS 2

1 THE 3.10

TOTAL 3.10

	HT	TVA	TTC
TVA 10.00 %	2.82	0.28	3.10

ESPECES 3.10

1 ARTICLE
(PASSAGE)

SAMEDI 04-02-2017 13:27:03
Cle 2-Serv.: 71-CAISSE 67-NOTE 000248/1

Montants exprimés en euros
Faites-nous part de vos remarques à
serviceclient@areas.com
Merci de votre visite et au revoir
Areas

Evaluation Metrics

Longest Common Subsequence Based Method

LCS score = length of LCS / length of ground truth

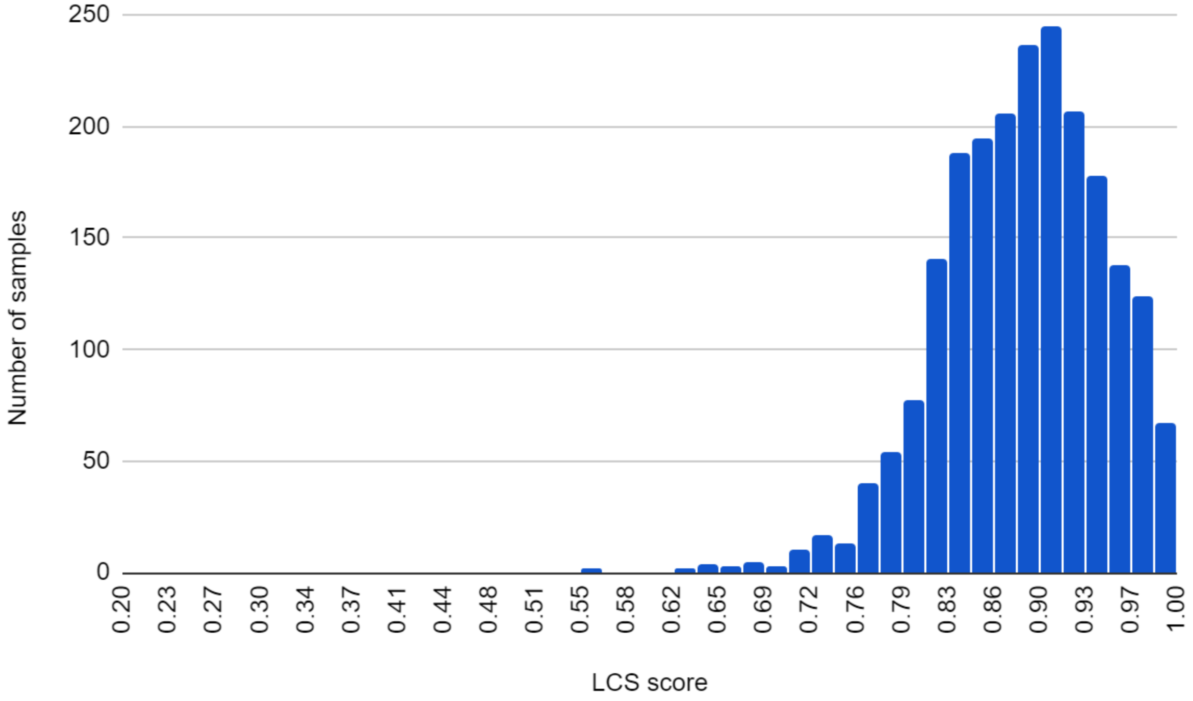
Bi-Partite Matching Based

BPM score = number of matched words/ number of words in the ground truth

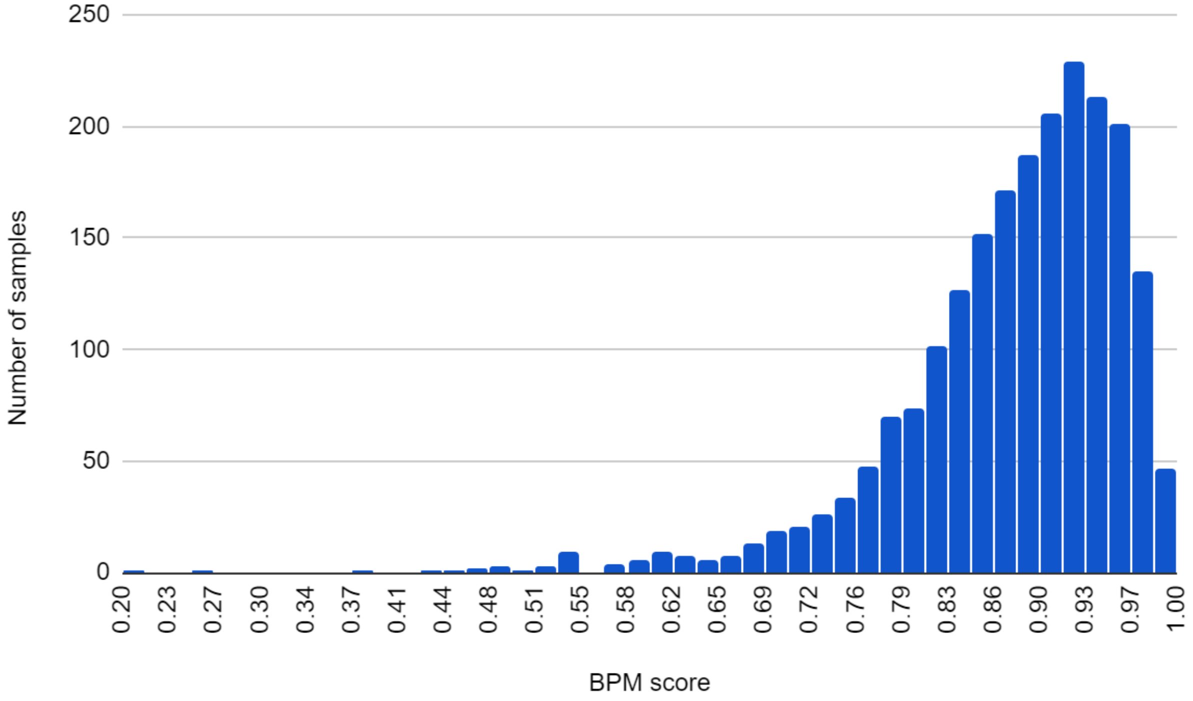
Evaluation Metrics

Ground truth	Test string	LCS	BPM
Thank You! Please come again. Keep the invoice for applicable returns.	*hank You! Please come again. Keep the invoice for applicable returns.	0.99	0.91
Thank You! Please come again. Keep the invoice for applicable returns.	***** You! Please come again. Keep the invoice for applicable returns.	0.93	0.91
Thank You! Please come again. Keep the invoice for applicable returns.	*hank*You!*Please*come*again. Keep the invoice for applicable returns.	0.93	0.55
Thank You! Please come again. Keep the invoice for applicable returns.	***** **** ****** **** *****. Keep the invoice for applicable returns.	0.66	0.55
Thank You! Please come again.	*hank You! Please come again.	0.97	0.80
Thank You! Please come again.	***** You! Please come again.	0.83	0.80
Thank You! Please come again.	*hank*You!*Please*come*again.	0.83	0.0
Thank You! Please come again.	Please come again. Thank You!	0.63	1.0

Evaluating the API

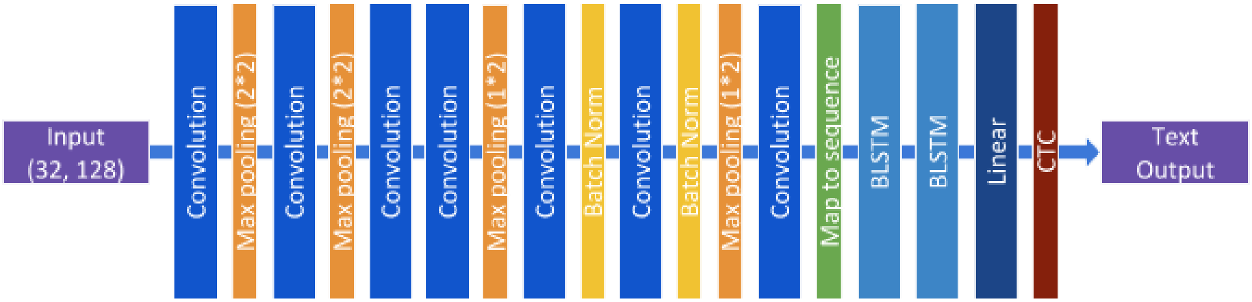


Evaluating the API

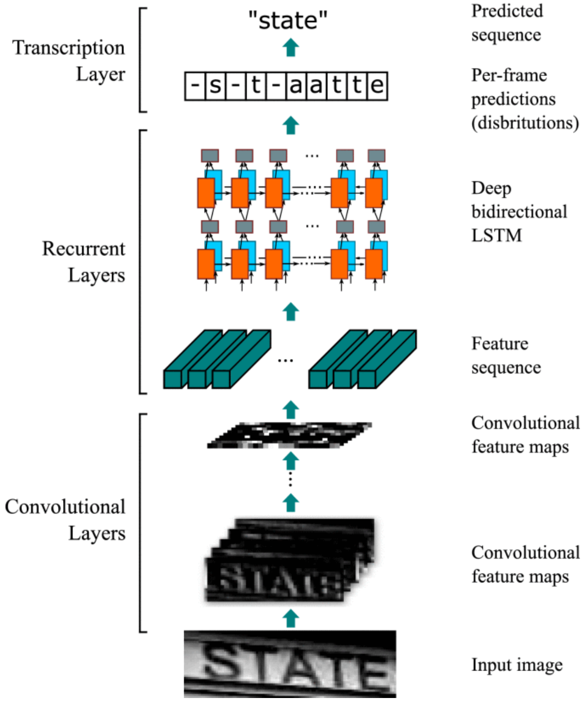


Simulator (approximator): CRNN model

Text Recognition



Convolutional Recurrent Neural Networks (CRNN)



Simulator

New dataset

- The CRNN model only accepts lines of text.
- Used the bounding boxes generated by Google API to crop the words from receipts.
- Expanded the dataset by adding 50000 sample from VGG synthetic text images dataset
- After cleaning the problematic images final training dataset was 247844 samples, and the test dataset contained 19981 samples

Training Modes

Jacovi et al. (2019) divide the training process of black-box approximator into three categories

- Offline training- Training examples are sampled from an apriori black-box input distribution.
- Online training- Output of the preprocessor is used to train the CRNN model.
- Hybrid training- CRNN model is trained for a few epochs and then it is trained in the online training setting.

Training Algorithms

- Nguyen and Ray (2020) suggest adding random noise to the input of the simulator model to enable it to explore more with input.
- They proposed an online training algorithm with this technique.

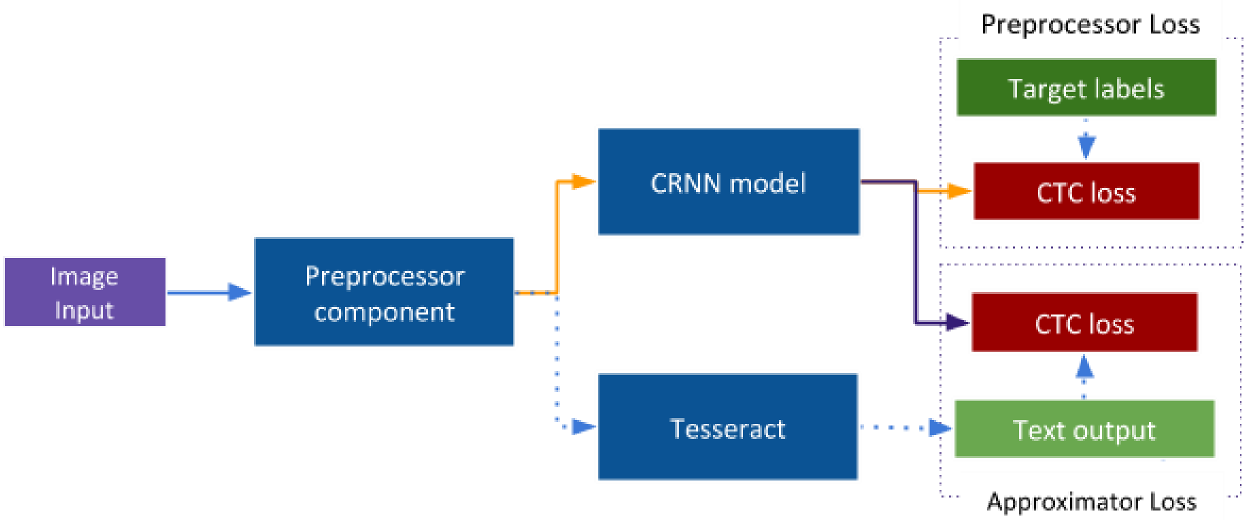
Algorithm 1: Adapted Online Training

for $J, p_{gt} \in Training\ \{Image,\ Ground\ Truth\}\ batch$ **do**
  $g = Preprocessor(J)$;
  do
    intialize s to 0;
    while s < S **do**
      sample std from $\mathcal{U}(\{0, 0.01, 0.02, 0.03, \ldots 0.09\})$;
      sample ϵ_s from $\mathcal{N}(0, std)$;
      $min_\phi \mathcal{L}(CRNN(g + \epsilon_s),\ Tesseract(g + \epsilon_s))$;
      $s = s + 1$
    end
  end;
  $min_\psi \mathcal{L}(CRNN(g), p_{gt})$;
end

Algorithm 2: Offilne Training

for $J, p_{gt} \in Training\ \{Image,\ Ground\ Truth\}\ batch$ **do**
  do
    intialize s to 0;
    while $s < S$ **do**
      sample std from $\mathcal{U}(\{0, 0.01, 0.02, 0.03, \ldots 0.09\})$;
      sample ϵ_s from $\mathcal{N}(0, std)$;
      $min_\phi \mathcal{L}(CRNN(J + \epsilon_s), Tesseract(J + \epsilon_s))$;
      $s = s + 1$
    end
  end;
  $g = Preprocessor(J)$;
  $min_\psi \mathcal{L}(CRNN(g), p_{gt})$
end

Training Pipeline



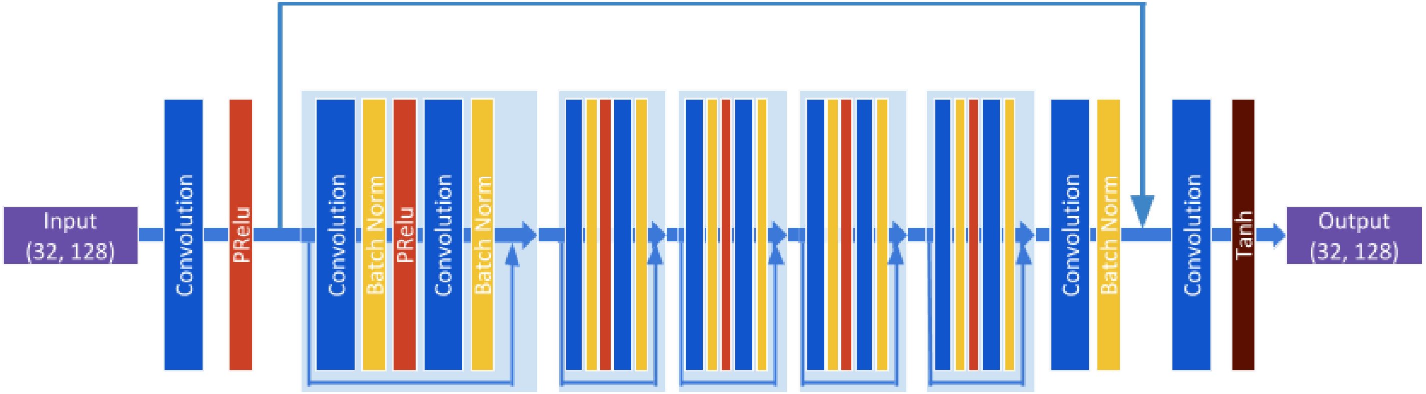
CRNN uses Connectionist Temporal Classification (CTC) loss

Preprocessor Models: Model 1

- We started with a model inspired by REDNet architecture.
- Models were trained in a hybrid training setting with offline algorithm with S (inner loop size) equal to 2.
- Tesseract yielded an accuracy of 70.48% (14082 correct words out of 19981 words) by using processed images.
- Using the original images, Tesseract only yielded an accuracy of 68.61%.

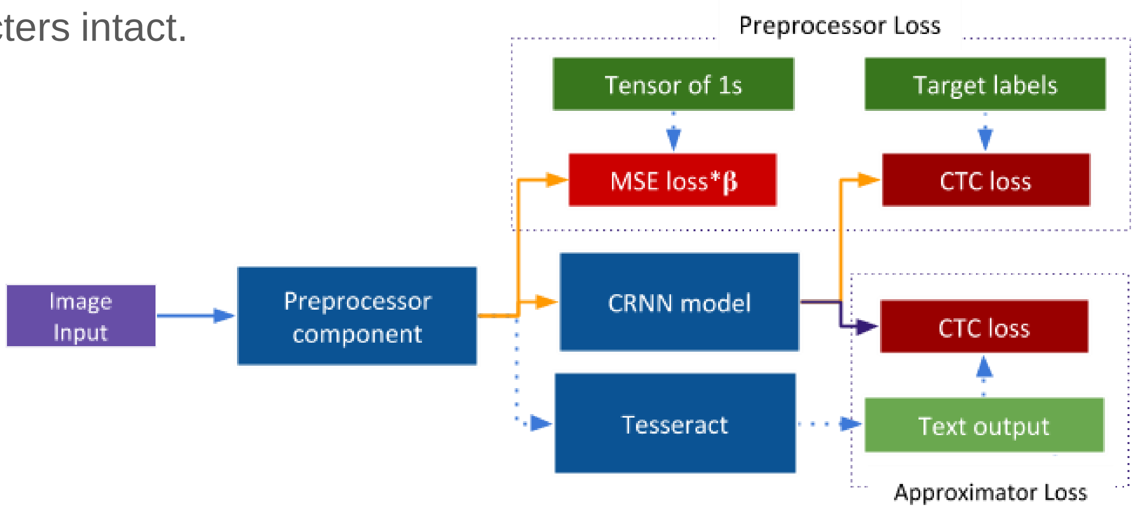
Preprocessor Models: Model 3

- With the success of REDNet inspired model we moved to a model adapted from generative model of SRGAN.
- This model was trained in the same setting with S equal to 5.
- Tesseract yielded an accuracy of 75.11% (15008 correct words out of 19981 words) with processed images.



Training with MSE Loss

- Document printed on a white background contains large white areas except for the printed text. This property can be used to clean the background leaving the important text pixels.
- While MSE loss nudges the output to be white, CTC loss pushed it leave the characters intact.



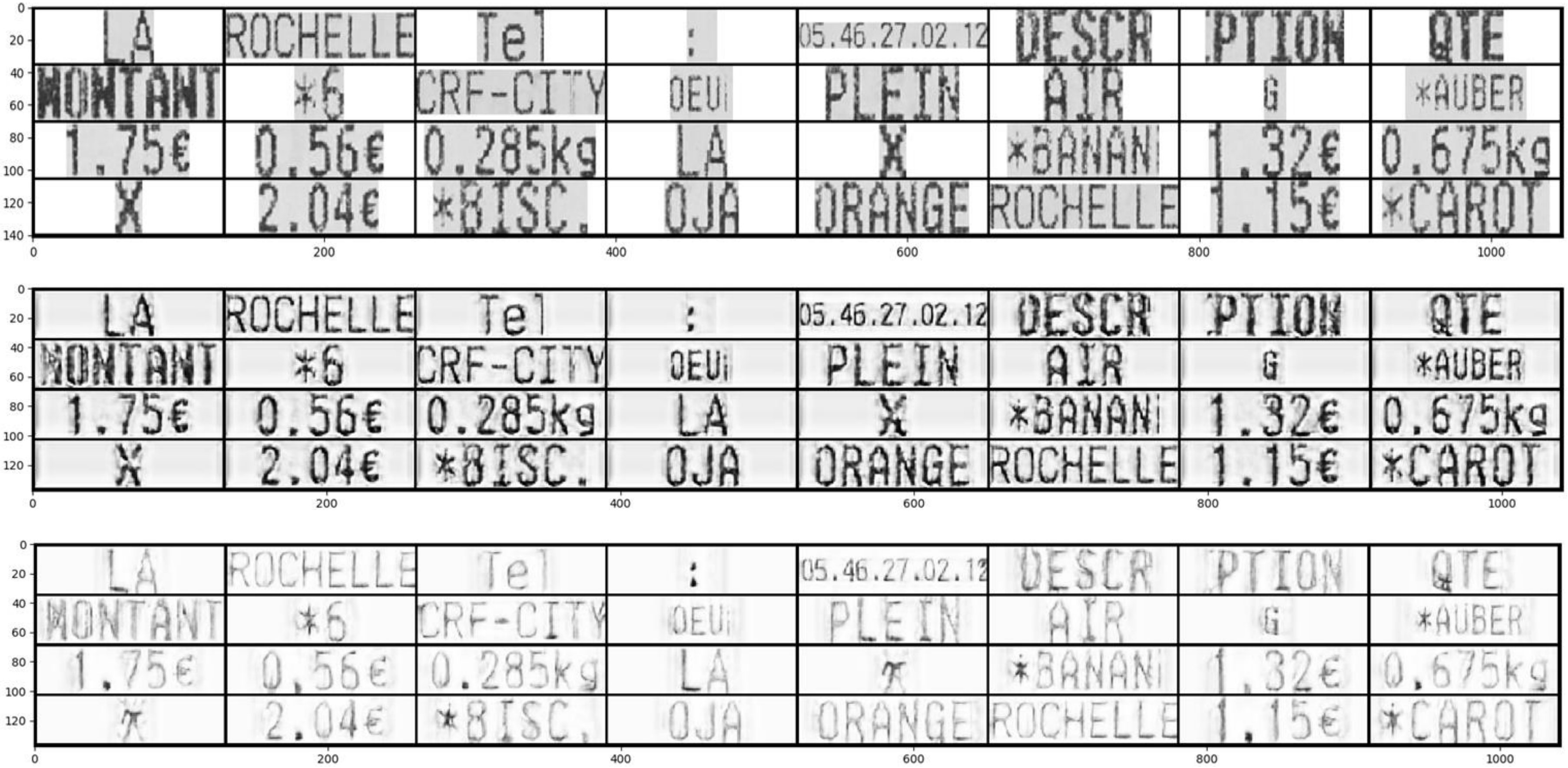
Preprocessor Models: Model 3

With MSE Loss

- Model 3 was trained in the previously mentioned pipeline and with its output, Tesseract yielded an accuracy of 78.11% (15607 correct words out of 19981 words).

With MSE Loss and Online Training

- Model 3 was trained with online training algorithm using similar settings.
- With preprocessed images, Tesseract performed with an accuracy of 81.27% (16238 correct words out of 19981 words).


LA
ROCHELLE
Tel
:
05.46.27.02.12
DESCR
PTION
QTE
MONTANT
*6
CRF-CITY
OEUi
PLEIN
AIR
G
*AUBER
1.75€
0.56€
0.285kg
LA
X
*BANANI
1.32€
0.675kg
X
2.04€
*8ISC.
OJA
ORANGE
ROCHELLE
1.15€
*CAROT

Results Summary

	Correct count	Word correctness
Unprocessed	13708	68.61%
Model 1 processed	14082	70.48%
Model 2 processed	13765	68.89%
Model 3 (trained with 12000 images) processed	14523	72.68%
Model 3 (trained with full dataset) processed	15008	75.11%
Model 3 (trained with MSE loss) processed	15607	78.11%
Model 3 (trained with online training) processed	16238	81.27%

Training with Image Patches

- Even though models are trained with cropped out images of words, in inference time we wanted to used them with full size images.
- Model output with full size images was not successful as expected.
- Therefore, we decided to train the model with image patches containing multiple words.
- In our pipeline, CRNN model has no capability of text detection. To used same pipeline we need coordinates of bounding boxes for text areas.

Image Patch Dataset

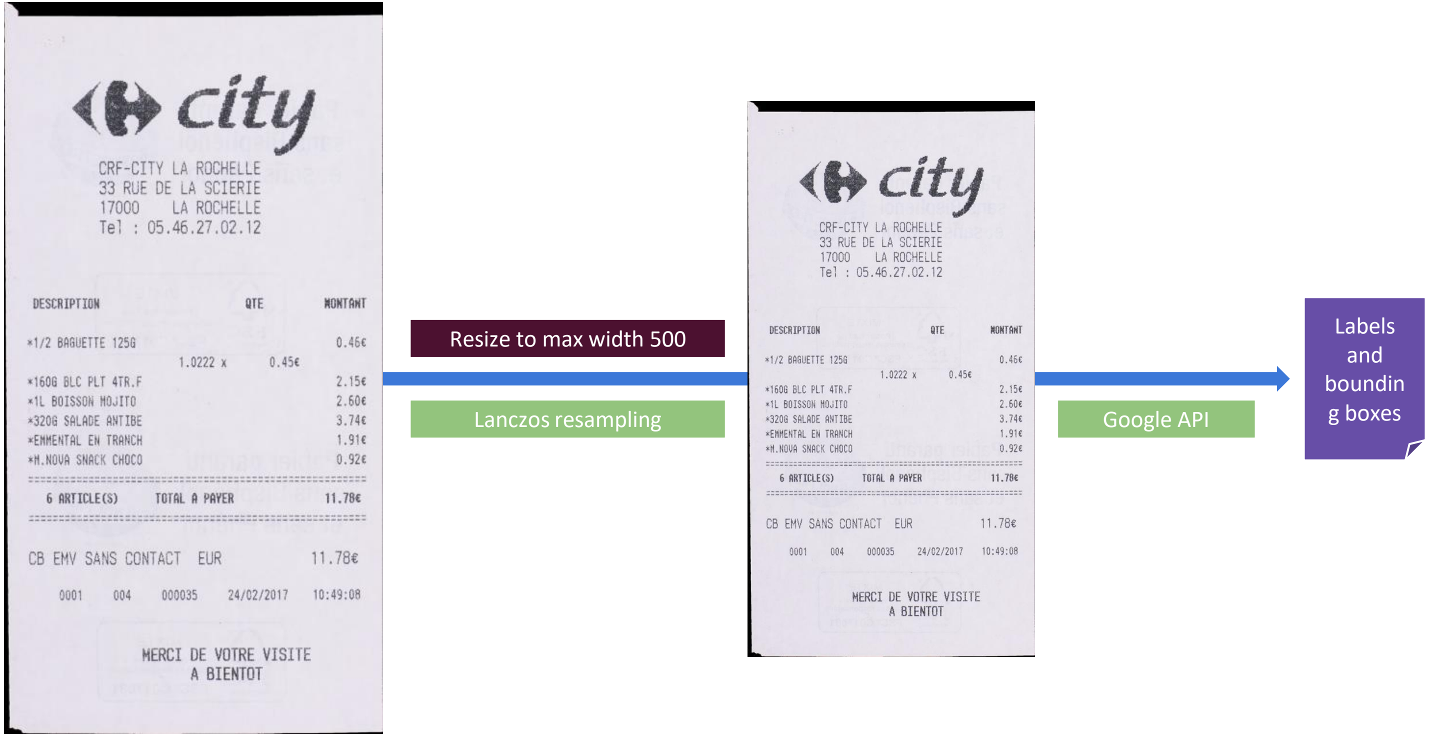


Image Patch Dataset

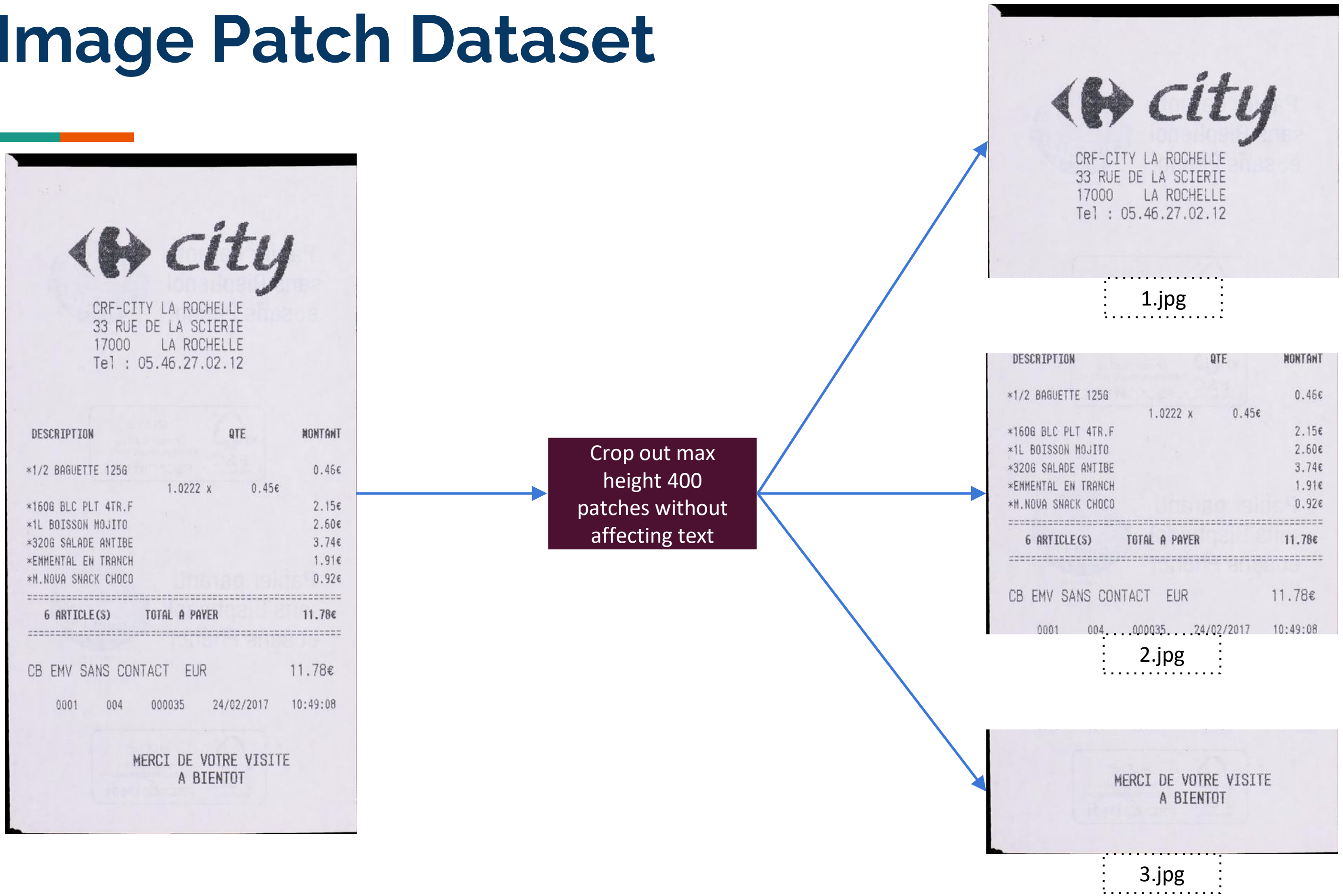
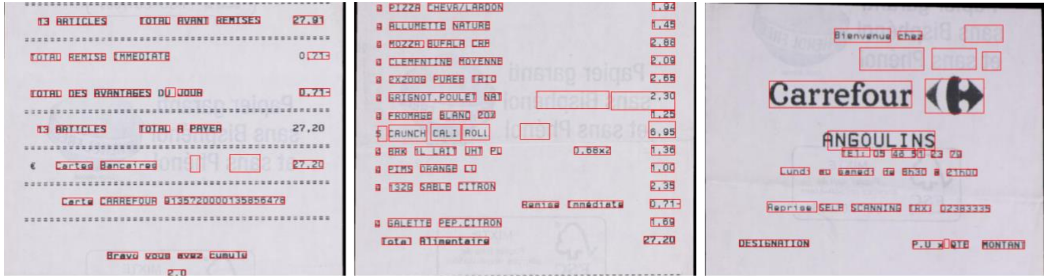
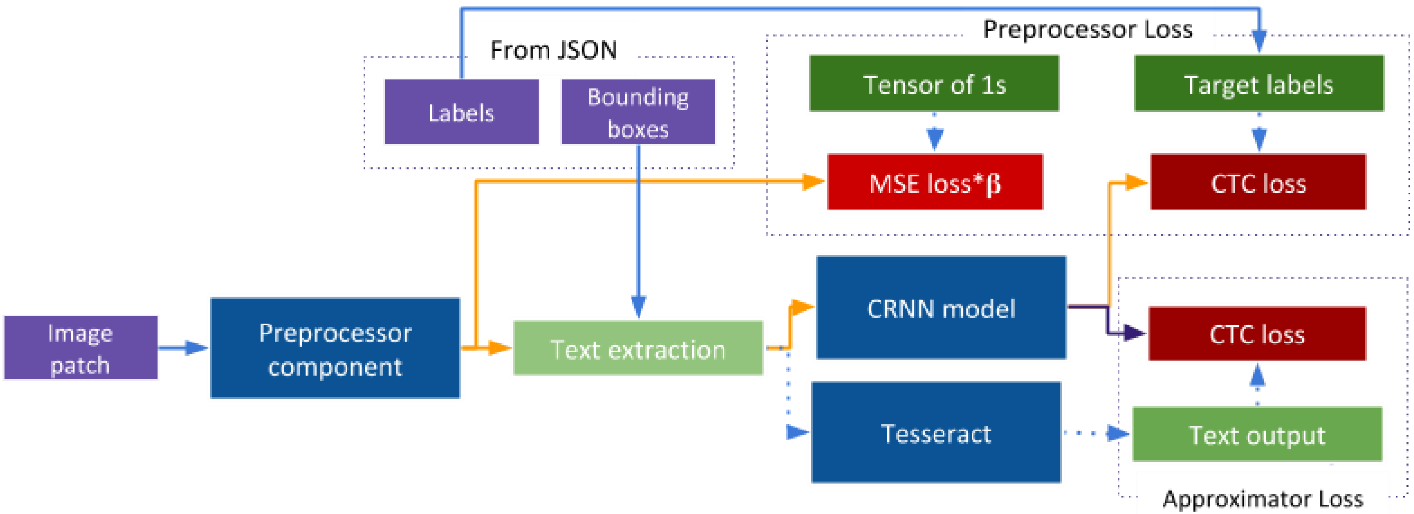


Image Patch Dataset

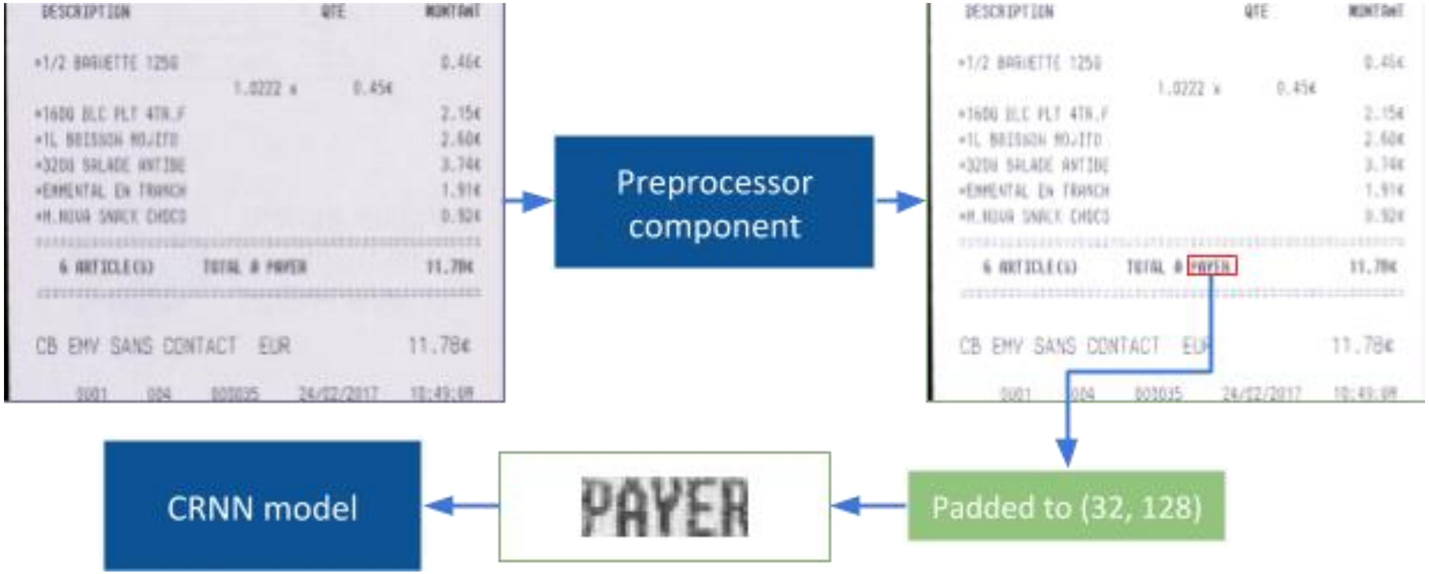
- Adjusted bounding boxes and labels for each patch is stored in separate JSON files.
- One problem with Google bounding boxes was some of them were shifted to right of the text area. In addition to that Google has detected some text printed in the back side of the receipt.
- We correct these errors by manually shifting the bounding boxes and removing garbage areas.



Training with Image Patch Dataset

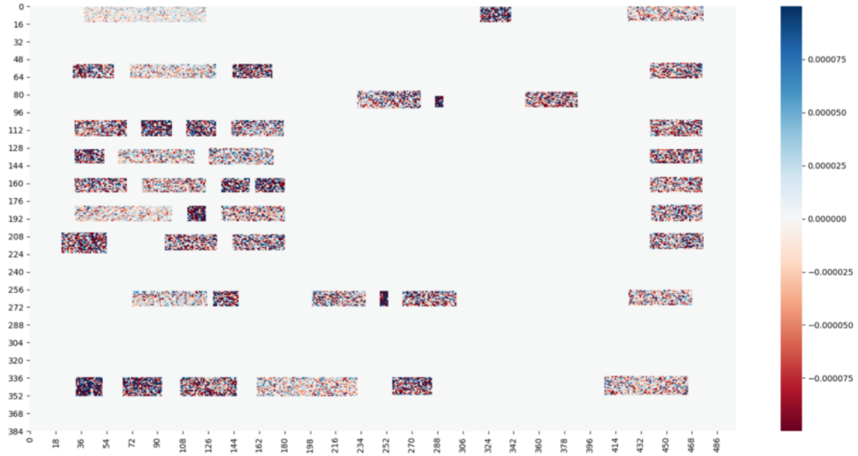


Training with Image Patch Dataset



Training with Image Patch Dataset

- Same model 3 was trained with image patches using similar settings with S equal to 1.
- Batch size for the preprocessor was one and batch size for CRNN model varied depending on the number of words in the image patch.



	Word correctness	LCS	BPM
Unprocessed images	58.24%	0.9	0.71
Processed images	88.94%	0.92	0.72

Observations

- Our final model was able boost Tesseract accuracy by 30% in word level. However it marginally improved the LCS and BPM score in paragraph level.
- The lucrative aspect of this type of pixel-level changes is that it can be used to create a preprocessor tailor-made for the particular OCR component.
- Even though the technique we used appeared simple and direct in the beginning, it was made clear that a lot of effort needs to be put into connecting different components.

References

Artaud, C. *et al.* (2018) ‘Find it! Fraud Detection Contest Report’, in *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 13–18.

Bojanowski, P. *et al.* (2017) ‘Enriching Word Vectors with Subword Information’, *Transactions of the Association for Computational Linguistics*, pp. 135–146. doi: 10.1162/tacl_a_00051.

Graves, A. *et al.* (2006) ‘Connectionist temporal classification’, *Proceedings of the 23rd international conference on Machine learning - ICML ’06*. doi: 10.1145/1143844.1143891.

Graves, A. *et al.* (2008) ‘Unconstrained On-line Handwriting Recognition with Recurrent Neural Networks’, in Platt, J. C. et al. (eds) *Advances in Neural Information Processing Systems 20*. Curran Associates, Inc., pp. 577–584.

Graves, A. and Schmidhuber, J. (2005) ‘Framewise phoneme classification with bidirectional LSTM and other neural network architectures’, *Neural Networks*, pp. 602–610. doi: 10.1016/j.neunet.2005.06.042.

Huang, Z. *et al.* (2019) ‘ICDAR2019 Competition on Scanned Receipt OCR and Information Extraction’, in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1516–1520.

Jacovi, A. *et al.* (2019) ‘Neural network gradient-based learning of black-box function interfaces’, in *International Conference on Learning Representations*. Available at: http://arxiv.org/abs/1901.03995.

Jaderberg, M. *et al.* (2016) ‘Reading Text in the Wild with Convolutional Neural Networks’, *International Journal of Computer Vision*, pp. 1–20. doi: 10.1007/s11263-015-0823

References

Ledig, C. *et al.* (2017) ‘Photo-realistic single image super-resolution using a generative adversarial network’, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690.

Le, Q. V. and Mikolov, T. (2014) ‘Distributed Representations of Sentences and Documents’, *arXiv [cs.CL]*. Available at: http://arxiv.org/abs/1405.4053.

Mao, X., Shen, C. and Yang, Y.-B. (2016) ‘Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections’, in Lee, D. D. et al. (eds) *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., pp. 2802–2810.

Mikolov, T. *et al.* (2013) ‘Efficient Estimation of Word Representations in Vector Space’, *arXiv [cs.CL]*. Available at: http://arxiv.org/abs/1301.3781.

Nguyen, N. M. and Ray, N. (2020) ‘End-to-end Learning of Convolutional Neural Net and Dynamic Programming for Left Ventricle Segmentation’, in *Medical Imaging with Deep Learning*.

Pennington, J., Socher, R. and Manning, C. (2014) ‘Glove: Global Vectors for Word Representation’, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. doi: 10.3115/v1/d14-1162.

Shi, B., Bai, X. and Yao, C. (2017) ‘An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition’, *IEEE transactions on pattern analysis and machine intelligence*, 39(11), pp. 2298–2304.