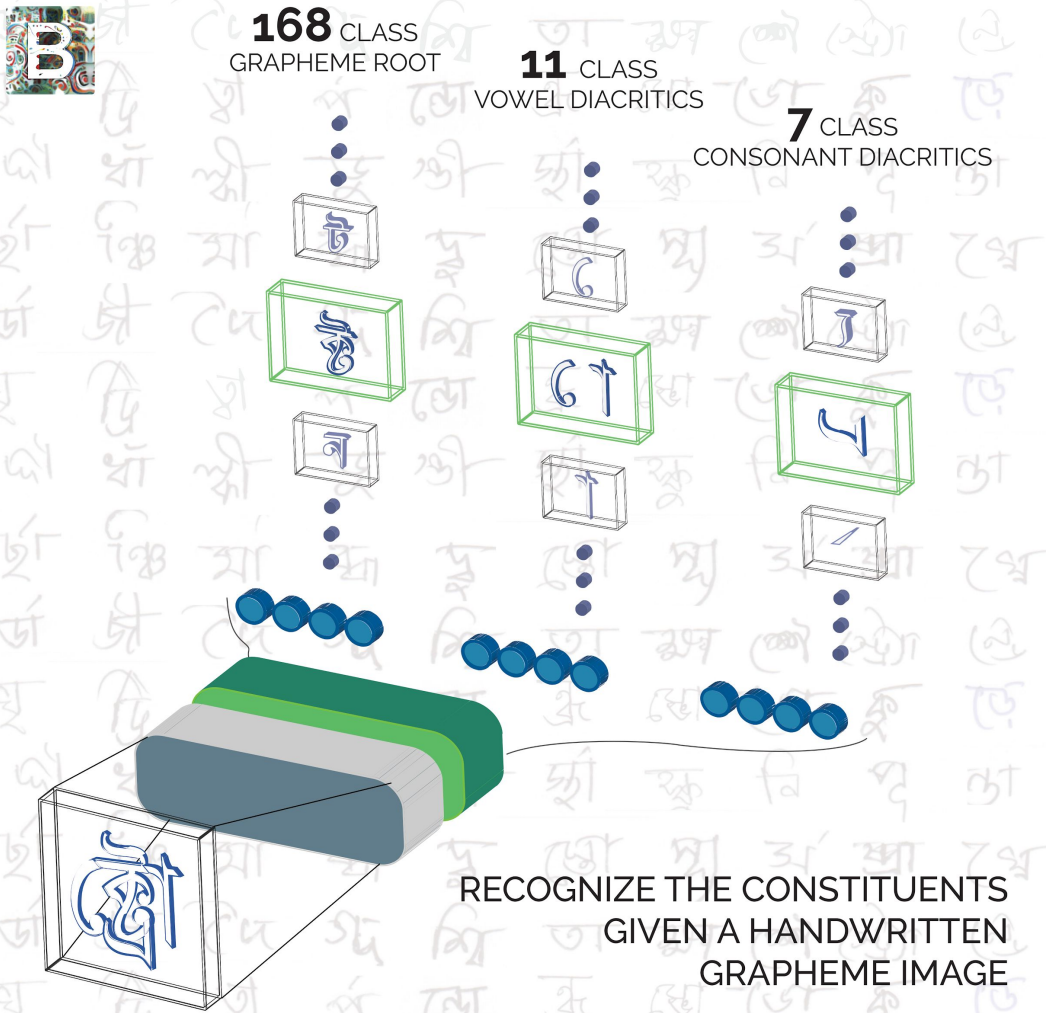
Checkoff-3 [70%]: Final System Prototype, Deliverable and Experimental results

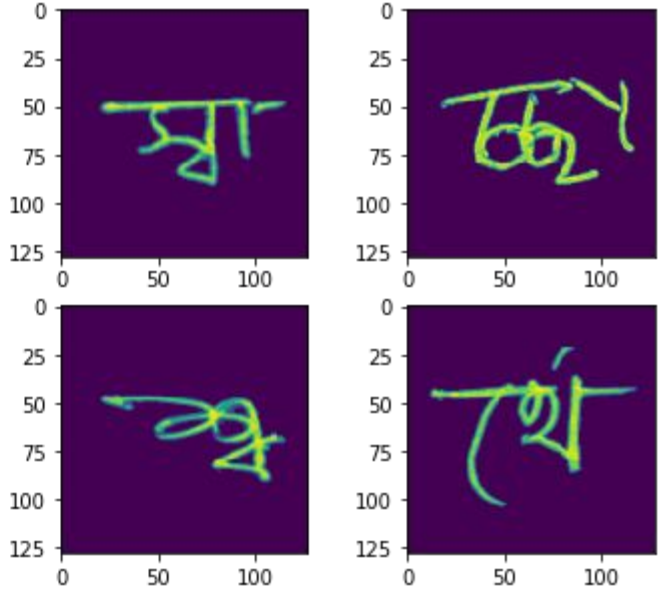
Deliverable / Grading item	Grading criteria	Remark
Novelty (15%)	• How novel is the proposed solution and system?	
System design and technical strength (30%)	• To what extent does the prototype apply Computer vision concepts and algorithms? • To what extent does the team design and implement computer vision concepts and algorithms	
Experiment and evaluation (15%)	• To what extent does the team evaluate appropriate computer vision algorithms for their problems?	
Report and poster (10%)	• Are the report and poster well written, well structured, concise and clear?	
Items checklist	• Presentation slides • Poster • Runnable source code • Experiment results • Report	

Bengali.AI Handwritten Grapheme Classification

Agenda

1. Problem Statement
2. Dataset
3. Baselines
4. Novel Models: Single Model (Resnet34) with Longer Tails, Seen and Unseen Model
5. Experiments - Data Generation and Augmentation, training
6. Evaluation Metrics


168 CLASS
GRAPHEME ROOT
11 CLASS
VOWEL DIACRITICS
7 CLASS
CONSONANT DIACRITICS
RECOGNIZE THE CONSTITUENTS
GIVEN A HANDWRITTEN
GRAPHEME IMAGE


0
25
50
75
100
125
0
50
100

Each Character can only have one class of each label

Label	Class
Grapheme roots (168)	অ, আ, ই, ঈ, উ, ঊ, ঋ, এ, ঐ, ও, ঔ, ক, ক্ক, ক্ট, ক্ত, ক্ল, ক্ষ, ক্ষ্ণ, ক্ষ্ম, ক্স, খ, গ, গ্ধ, গ্ন, গ্ব, গ্ম, গ্ল, ঘ, ঘ্ন, ঙ, ঙ্ক,ঙ্ক্ত, ঙ্ক্ষ, ঙ্খ, ঙ্গ, ঙ্ঘ, চ, চ্চ, চ্ছ, চ্ছ্ব, চ্ছ্র, জ, জ্জ, জ্ঝ, জ্ঞ, জ্ব, ঝ, ঞ, ঞ্চ, ঞ্ছ, ঞ্জ, ট, ট্ট, ঠ, ড, ড়, ড্ড, ঢ, ঢ়, ণ, ণ্ট, ণ্ঠ, ণ্ড, ণ্ণ, ত, ত্ত, ত্ত্ব, থ, থ্ব, ত্ব, ত্ম, থ, দ, দ্ঘ, দ্দ, দ্ধ, দ্ব, দ্ভ, দ্ম, ধ, ধ্ব, ন, ন্জ, ন্ট, ন্ঠ, ন্ড, ন্ত, ন্ত্ব, ন্থ, ন্দ, ন্দ্ব, ন্ধ, ন্ন, ন্ব, ন্ম, ন্স, প, প্ট, প্ত, প্ন, প্প, প্ল, প্স, ফ, ফ্ট, ফ্ফ, ফ্ল, ব, ব্জ, ব্দ, ব্ধ, ব্ব, ব্ল, ভ, ভ্ল, ম, ম্ন, ম্প, ম্ব, ম্ভ, ম্ম, ম্ল, য, য়, র, ল, ল্ড, শ, শ্ব, ষ, স, স্ট, স্থ, হ, হ্ণ, হ্ন, হ্ট, হ্ন, হ্ব, হ্ম, হ্ল, হ্ল, শ্চ, শ্ন, শ্ম, শ্ল, ষ্ক, ষ্ট, ষ্ঠ, ষ্ণ, ষ্প, ষ্ফ, ষ্ম, স্ক, স্ত, স্ন, স্প, স্ফ, স্ব, স্ম, স্ল, স্স, হু, হৃ, হ্ম, হ্ল, ৎ, ং, ঃ
Vowel Diacritics (11)	Null, া, ি, ী, ু, ূ, ৃ, ে, ৈ, ো, ৌ
Consonant Diacritics (7)	Null, ্য (য-ফলা), ্র (র-ফলা), র্ (রেফ), ঁ, র্্য (রেফ য-ফলা), ্র্য (র-ফলা য-ফলা)

Problem Statement

Based on a given image of a handwritten Bengali grapheme, to separately classify three constituent elements in the image: ***grapheme root, vowel diacritics, and consonant diacritics.***

Dataset

- Provided data contains 1295 of the 12936 (168*11*7) possible graphemes.
- Private Test set contains graphemes that are unseen from the training set.
- Referencing Bengali Dictionary, we estimate that the Bengali language contains roughly 3000 unique graphemes, **but there were only 1295 graphemes in the training dataset**

Datasets - Training and validation splits

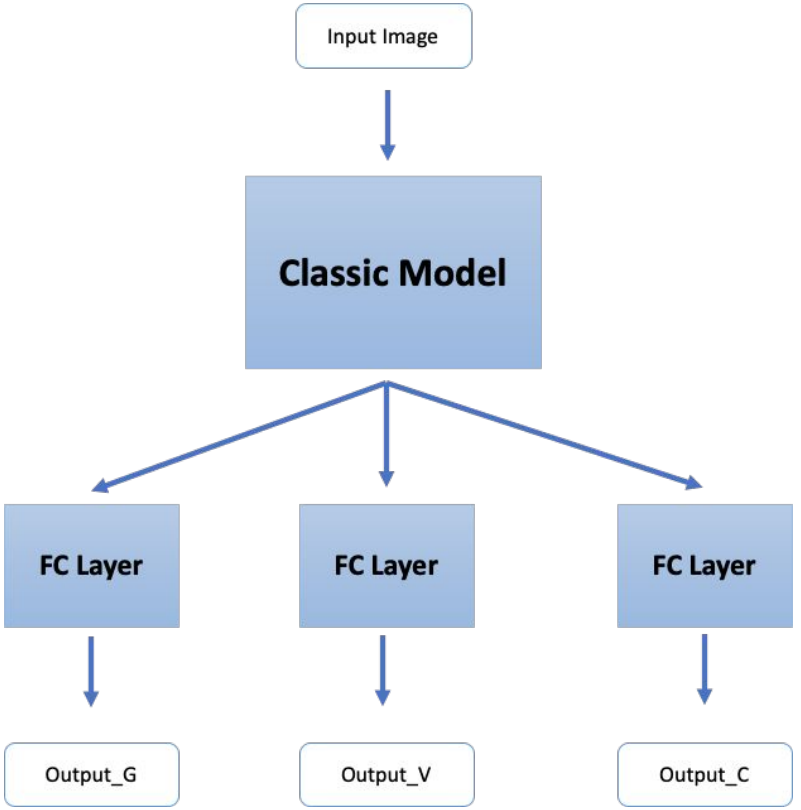
To simulate unseen data, a small sample of combinations are picked that are not present in the training set.

The final split are as follows:

- Training set: 77%
- Validation seen set: 20%
- Validation unseen set 3%

Baselines

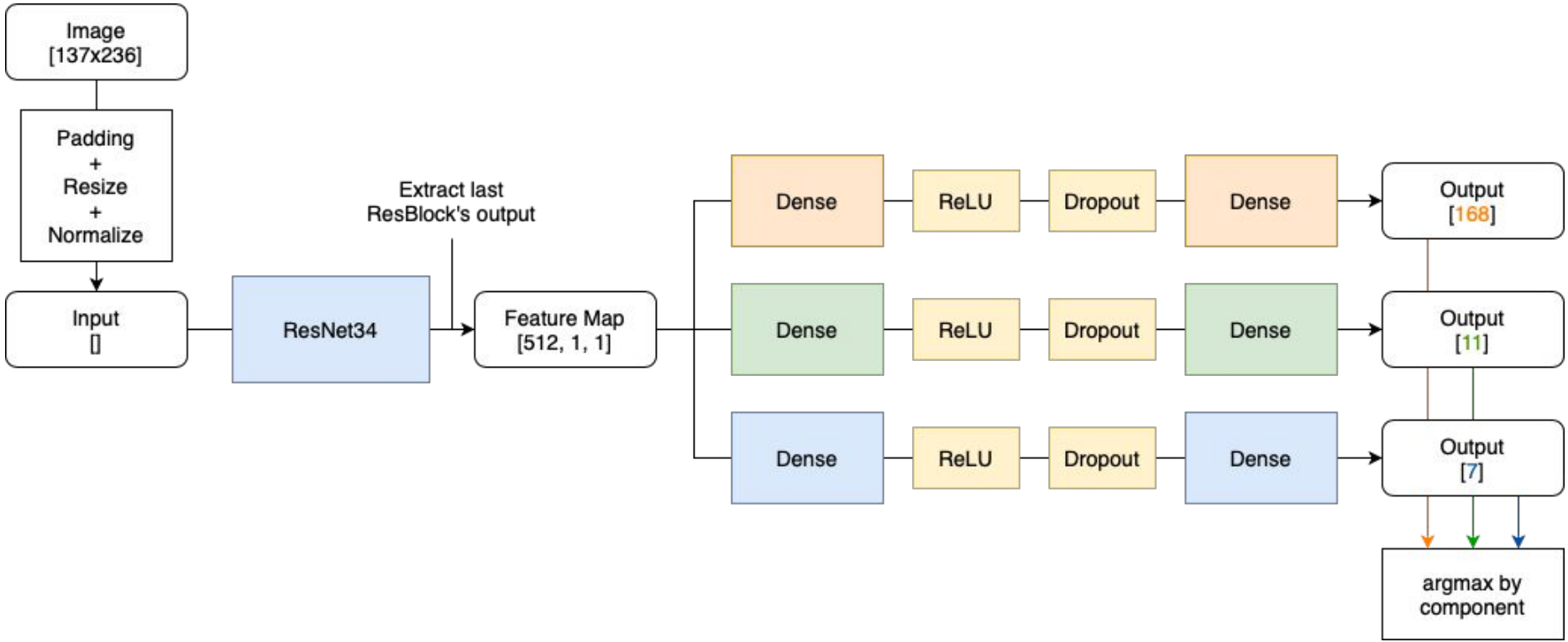
- Classical model + three tails output
- Transfer Learning - Fine tuned for 3 epochs
- Report accuracy on seen and unseen validation data



Baseline Performance

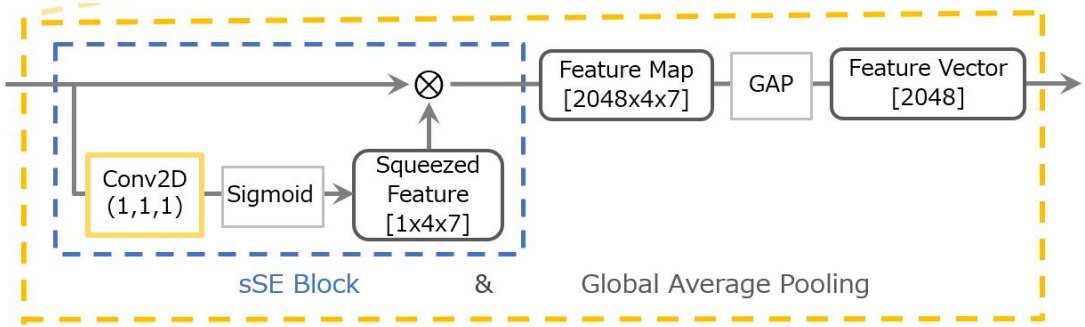
Model	Seen Accuracy	Unseen Accuracy
Resnet18	**0.9485**	0.8391
GoogleNet	0.9442	0.8285
MobileNet_V2	0.934	0.812
VGG16	0.9327	**0.8492**
AlexNet	0.8284	0.7205

Model - Single Model with Longer Tails



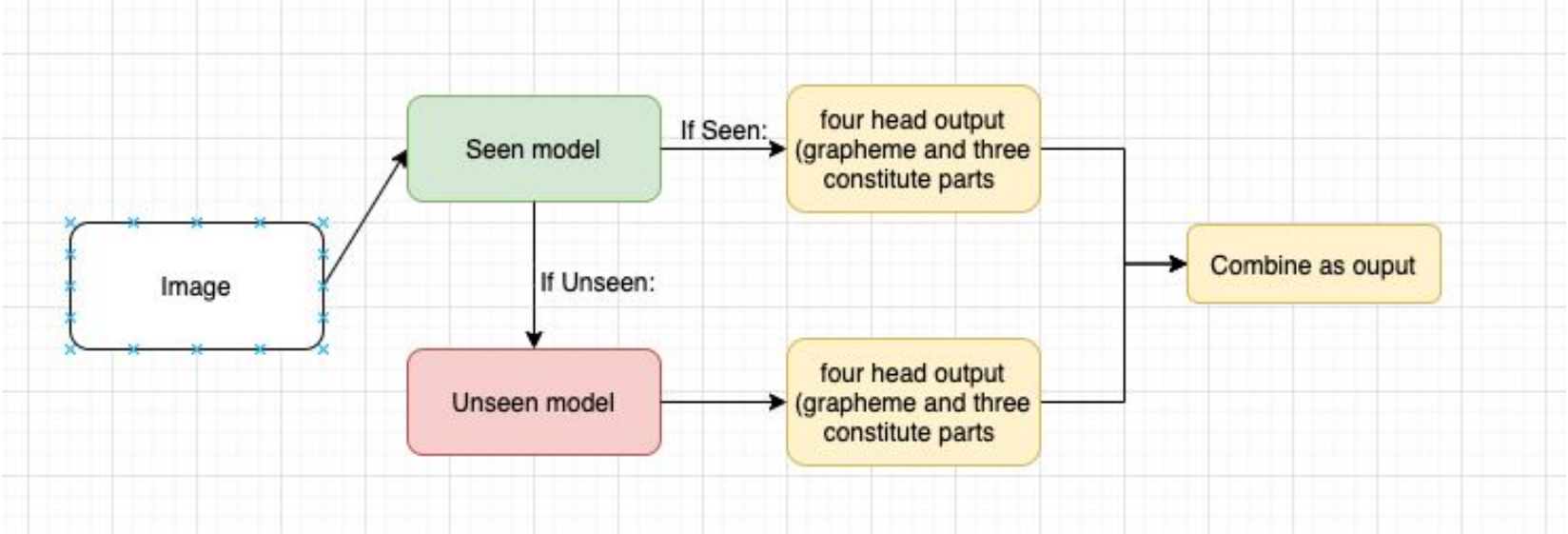
Model - Resnet34 with Longer Tails

- 3 tails model
  - Take the feature map from the last convolution block, pass it into three different fully connected blocks.
- sSE (Channel Squeeze and Spatial Excitation Block) Pooling
  - Focus on different parts of the image for different constituents



https://arxiv.org/pdf/1803.02579.pdf

Model - Seen and unseen model (hyperparameters in Appendix B)



Seen model VS Unseen model (see appendix C and appendix D)

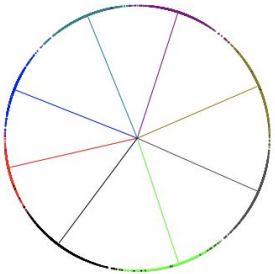
- Seen Model only uses very few augmentation to make sure it can ‘overfit’ to seen graphemes, while Unseen Model uses heavy augmentations to make it generalize to unseen ones.
- Both model output 4 parts: grapheme id, grapheme root, consonant and vowel. While Seen model has one extra output: Arcface loss.
- Seen model has different architecture as unseen model on the top layers. The reason is we want to separate the correlation with grapheme id and its three subcomponents

Seen and Unseen model - Arcface loss [6]

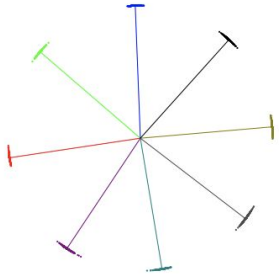
Usage: to distinguish the whether a grapheme is seen in the training dataset.

Procedure: we presume Arcface loss can indicate whether a grapheme is further distance from all the known clusters. Just like in face recognition.

$$L_3 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}. \quad (3)$$



(a) Softmax



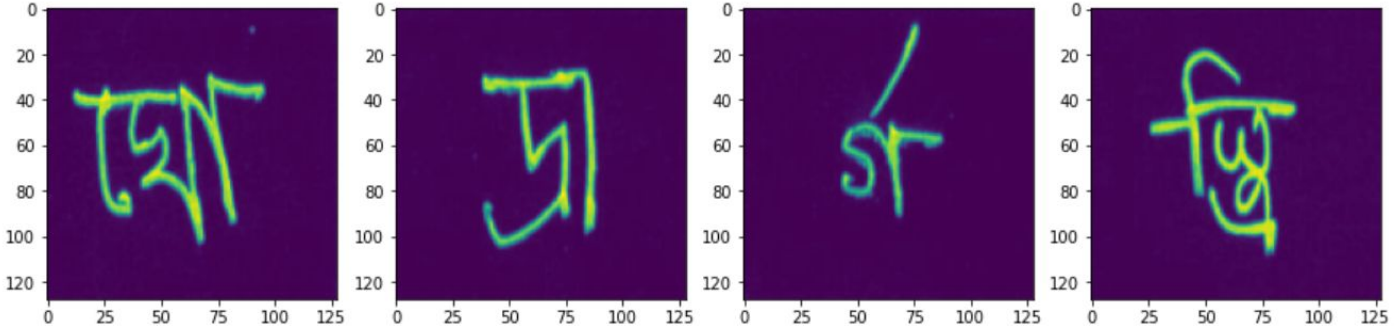
(b) ArcFace

Experiments - Data Generation

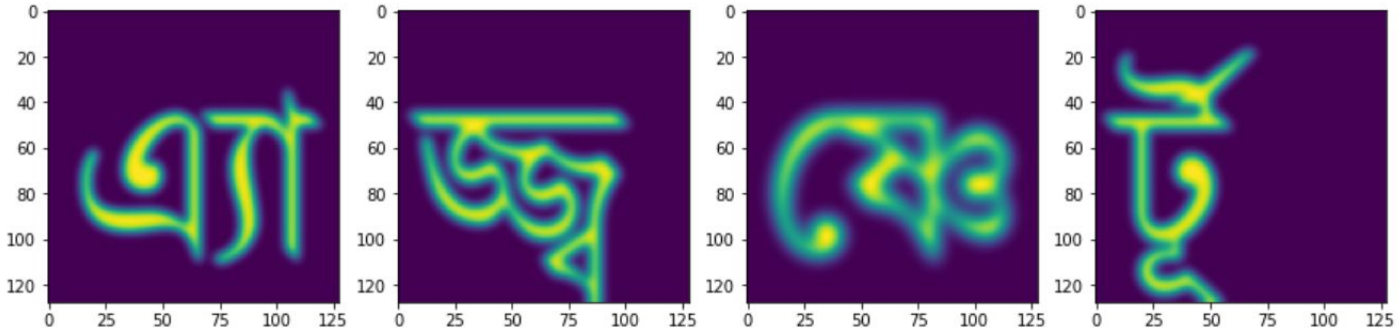
- Generate text as images using different fonts
- Apply gaussian blur to improve generalisation (edges are less sharp)
- Concatenate it with provided dataset for more training data on unseen graphemes
- Aim to recognize overall topology

Experiments - Data Generation

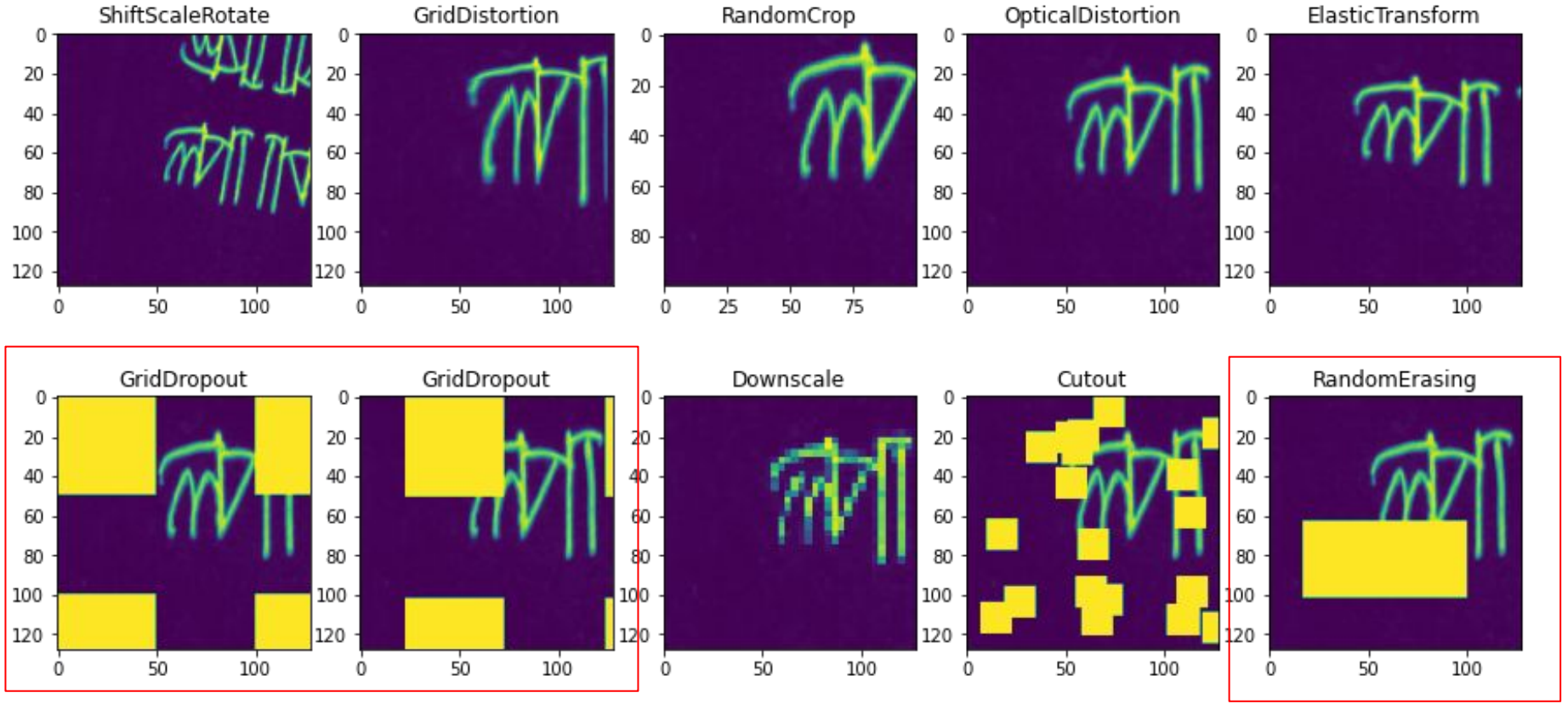
Provided:



Generated:



Experiments - Data Augmentation



Experiments - training details

Cutmix: Cuts random portions of the images and mixes it with other images. Loss function is a sum of weighted cross entropy loss from the combined images.

Dropout: we used dropout before the head layer for preventing overfitting

SGDR: we used the cosine scheduler from pytorch library for better convergence

Swish activation: a more powerful self gated activation function: x.sigmoid(x). This is proposed by google that claims to have better performance on deep models.

Experiments: Training details

Test augmentation:

Apply test augmentation is beneficial for predictions that enables parts of the model architecture.

K-fold evaluation:

We are evaluating our model on 5 - fold data to for hyperparameter tuning

Loss function:

for grapheme id, root and consonant the loss is softmax loss. Then it is combined with ArcFace Loss with their special weights by experiment.

Snapshot Ensemble:

Model is saved at various epochs and softmax outputs are combined to produce a better score

Experiments - Evaluation metric

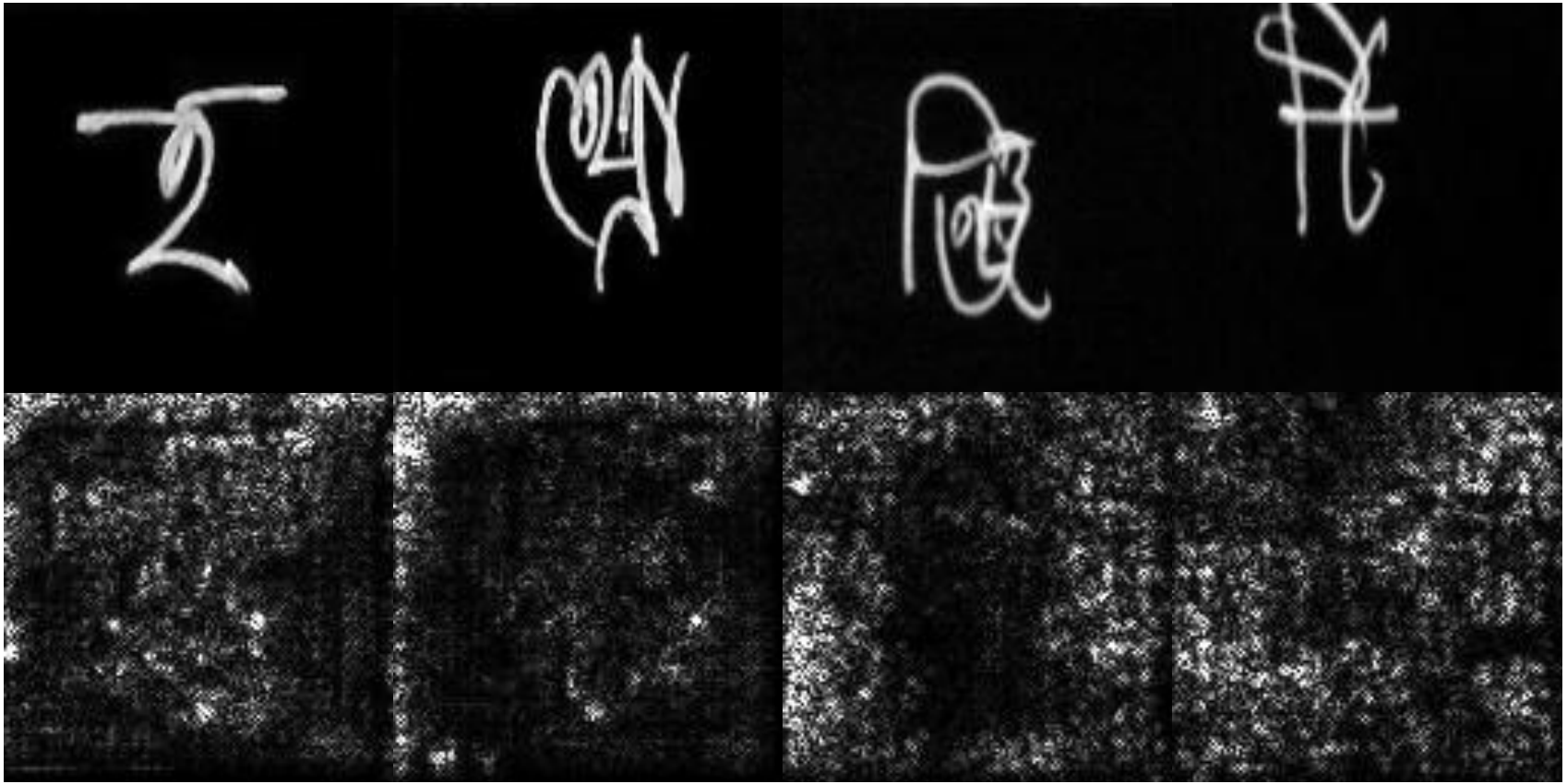
We followed the evaluation metric proposed by Kaggle[12]:

Averaged Macro recall of “root”, “consonant” and “vowel”, the score is combined with a weight of 2:1:1. (this is also reflected in our loss function)

Results

Model	Private Score	Public Score
ResNet18 Baseline	0.8926	0.9529
ResNet34LongerTail 34 epoch	0.9043	0.9605
ResNet34LongerTail 69 epoch	0.9027	0.9636
ResNet34LongerTail 104 epoch	0.9023	0.9641
ResNet34LongerTail SnapshotEnsemble	0.9084	0.9661
sSEResNetLongerTail SnapshotEnsemble	0.9247	0.9737
Seen and Unseen Model	0.9350	0.9814

Results - interpretation



Future Improvements

1. Use cyclic GAN to generate new images from typed graphemes that have similar styles with the dataset to solve the problem of unseen.
2. Only cut out pixels of different parts(root, consonant, vowel) of the grapheme instead of simple cutting out the whole grid just like we did in cutmix. This can be done using edge detection.
3. Attention-based extraction models for optical character recognition.[3]

References

[1]: https://arxiv.org/pdf/1905.04899v1.pdf

[2]: https://github.com/clovaai/CutMix-PyTorch

[3]: https://arxiv.org/pdf/1704.03549v4.pdf

[4]: https://zhuanlan.zhihu.com/p/60747096

[5]: https://zhuanlan.zhihu.com/p/62680658

[6]: https://arxiv.org/pdf/1801.07698.pdf

[7]: https://medium.com/1-minute-papers/arcface-additive-angular-margin-loss-for-deep-face-recognition-d02297605f8d

[8]: https://www.groundai.com/project/arcface-additive-angular-margin-loss-for-deep-face-recognition/1

[9]: http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

[10]: https://arxiv.org/pdf/1608.03983.pdf

[11]: https://arxiv.org/pdf/1710.05941.pdf

[12]: https://www.kaggle.com/c/bengaliai-cv19/overview/evaluation

Appendix A0: training parameters for baseline models

batch_size = 128

num_workers = 4

init_lr = 0.001

grapheme_dim = 1295

firstcomponent_dim = 168

secondComponent_dim = 11

thirdComponent_dim = 7

n_epochs = 3

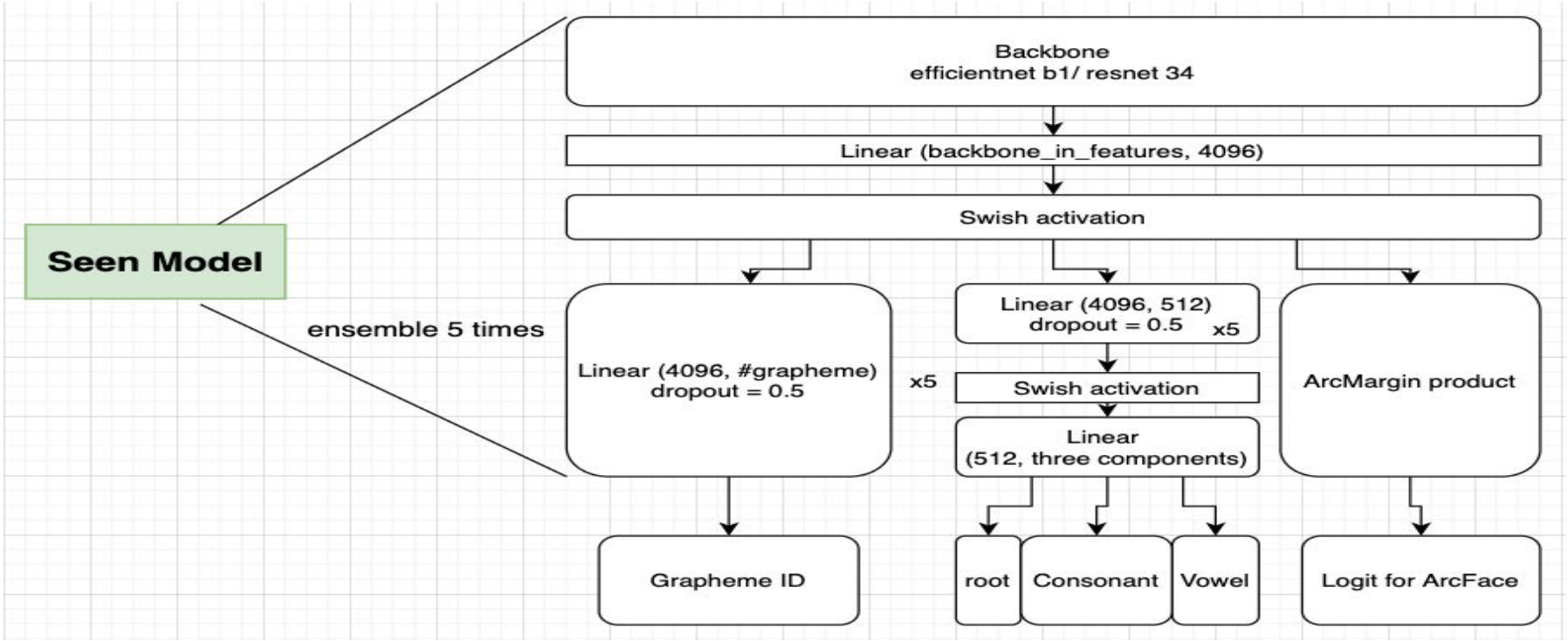
Pre_trained = True

Appendix A1 - training parameters for ensemble model

Appendix B: training parameters for seen and unseen models

```
enet_type = 'efficientnet-b1'

HEIGHT = 137

WIDTH = 236

image_size = 128

cut_size = int(image_size * 0.85)

data_dir = '../input/bengaliaicv19feather'

batch_size = 128

num_workers = 4

init_lr = 0.001

grapheme_dim = 1295

firstcomponent_dim = 168

secondComponent_dim = 11

thirdComponent_dim = 7

out_dim = c0_dim + c1_dim + c2_dim + c3_dim

loss_weight = [4, 2, 1, 1, 2]

n_epochs = 60

Kfold = 5
```

Appendix C - Seen model



Appendix D - Unseen model