

EENet: Learning to Early Exit for Adaptive Inference

Mateusz Pach

March 22, 2023

Original paper

EENET: LEARNING TO EARLY EXIT FOR ADAPTIVE INFERENCE

Fatih Ilhan, Ling Liu, Ka-Ho Chow, Wenqi Wei, Yanzhao Wu

School of Computer Science

Georgia Institute of Technology

Atlanta, GA, USA

{filhan, ll72, khchow, wenqiwei, yanzhaowu}@gatech.edu

Myungjin Lee, Ramana Kompella, Hugo Latapie, Gaowen Liu

CISCO Research

San Jose, CA, USA

{myungjle, rkompell, hlatapie, gaoliu}@cisco.com

ABSTRACT

Budgeted adaptive inference with early exits is an emerging technique to improve the computational efficiency of deep neural networks (DNNs) for edge AI applications with limited resources at test time. This method leverages the fact that

Early exit

Networks don't have to use their full potential to correctly predict easy examples.

Let's exit when we are confident enough.

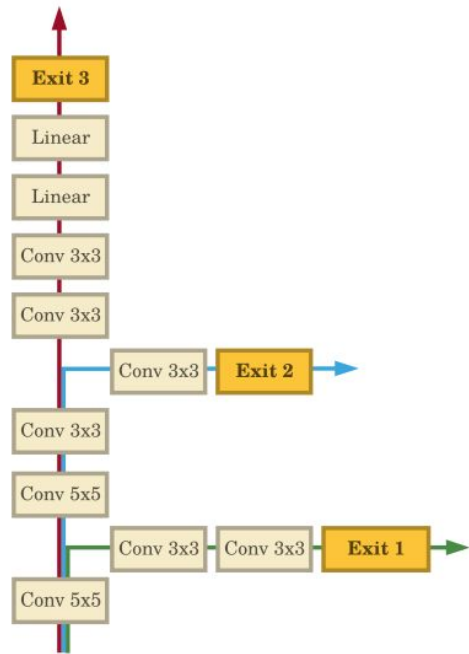


Fig. 1: A simple BranchyNet with two branches added to the baseline (original) AlexNet. The first branch has two convolutional layers and the second branch has 1 convolutional layer. The “Exit” boxes denote the various exit points of BranchyNet.

Goal

Learn optimal early exit policies independent of the multi-exit DNN training process given expected exit time.

Goal

Learn optimal early exit policies independent of the multi-exit DNN training process given expected exit time.

In other words,

- we don't touch the original parameters,
- we don't rely on number of exits,
- we don't set task-specific rules,
- we have constraint on average exit time,

and we answer how many and what examples should leave in each exit.

Architecture

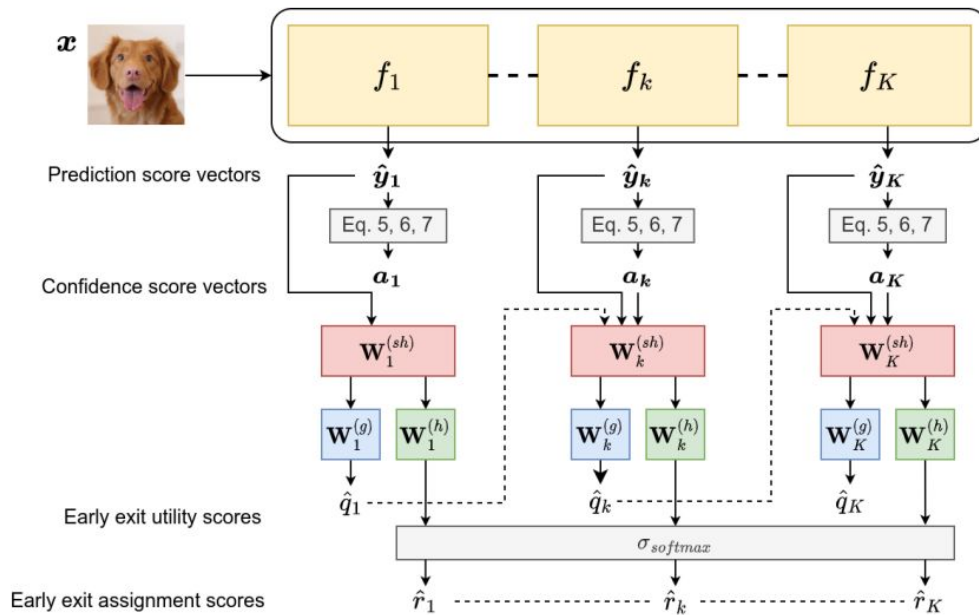


Figure 2: System architecture of EENet.

Architecture

These are segments of our original network.

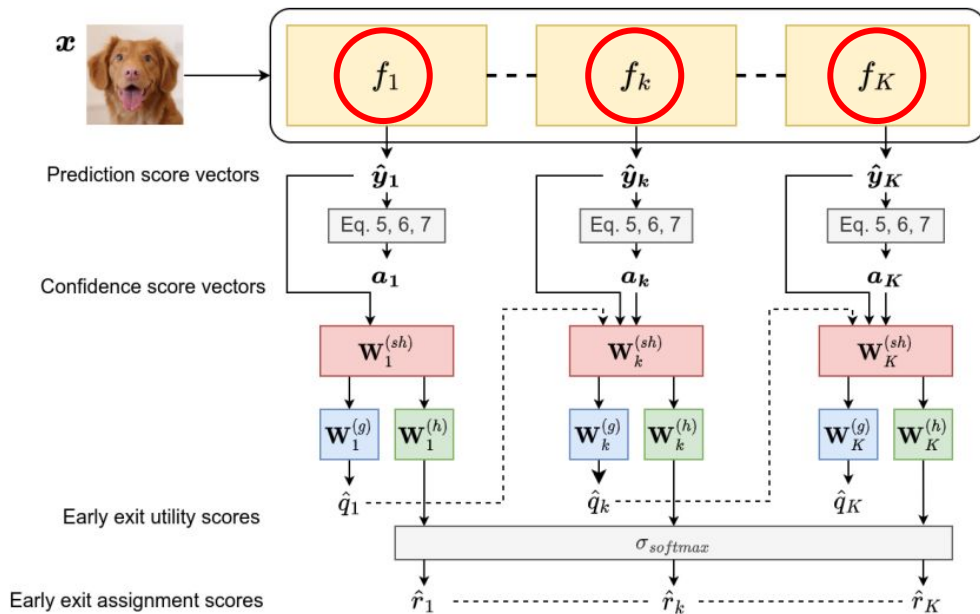


Figure 2: System architecture of EENet.

Architecture

Prediction score vectors are the outputs from the exits.

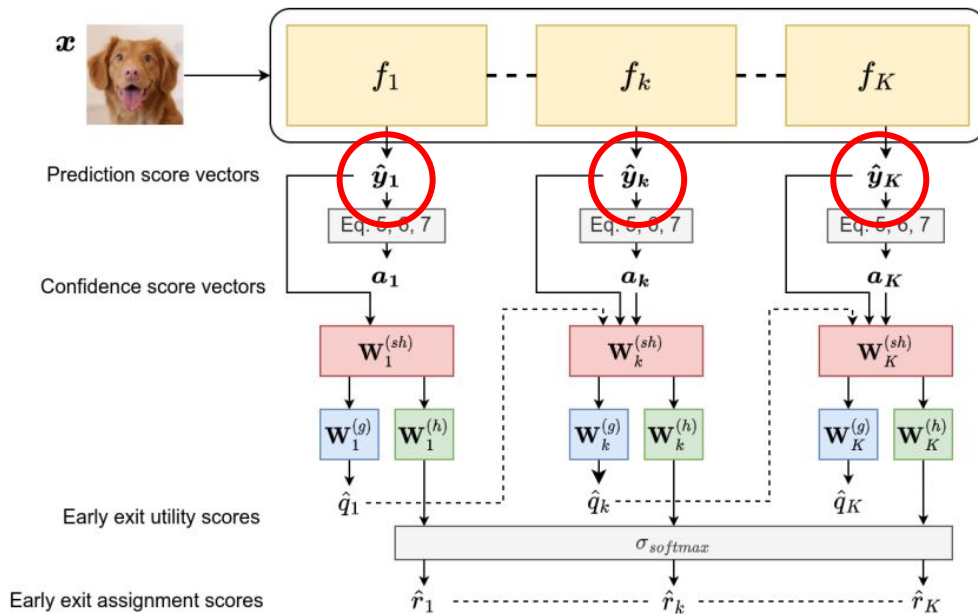


Figure 2: System architecture of EENet.

Architecture

Confidence score vectors are 3-factor confidence heuristic.

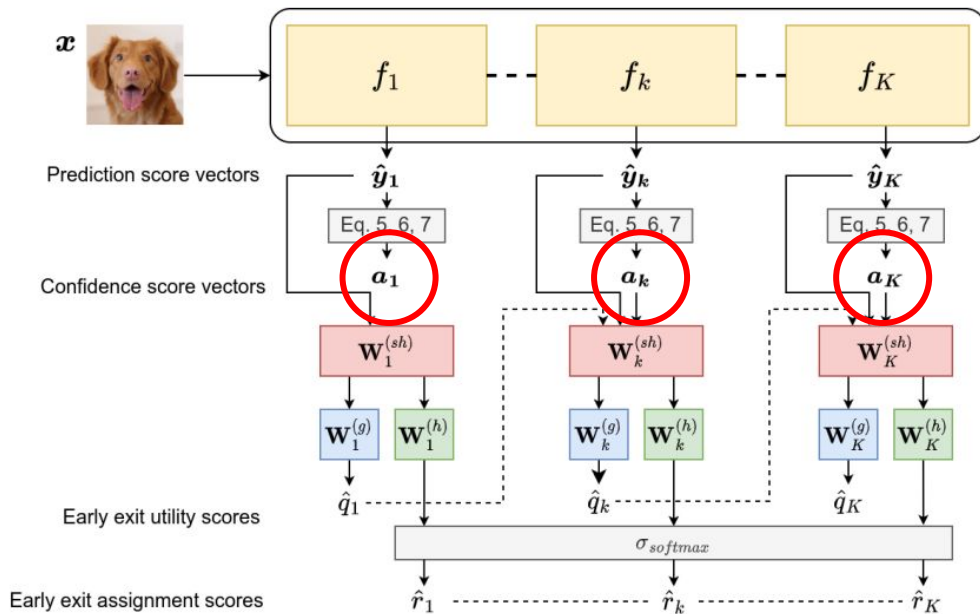


Figure 2: System architecture of EENet.

Architecture

These are fully connected layers computing our target variables.

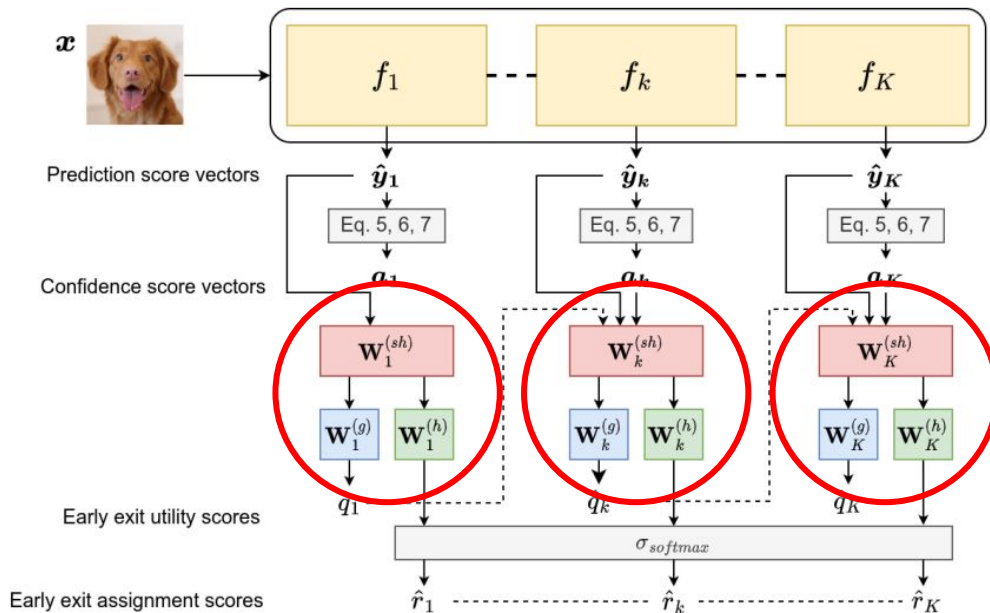


Figure 2: System architecture of EENet.

Architecture

Early exit utility scores are used to determine if we should exit. We use them instead of naive confidence score.

The higher it is, the more confident we are the output is correct.

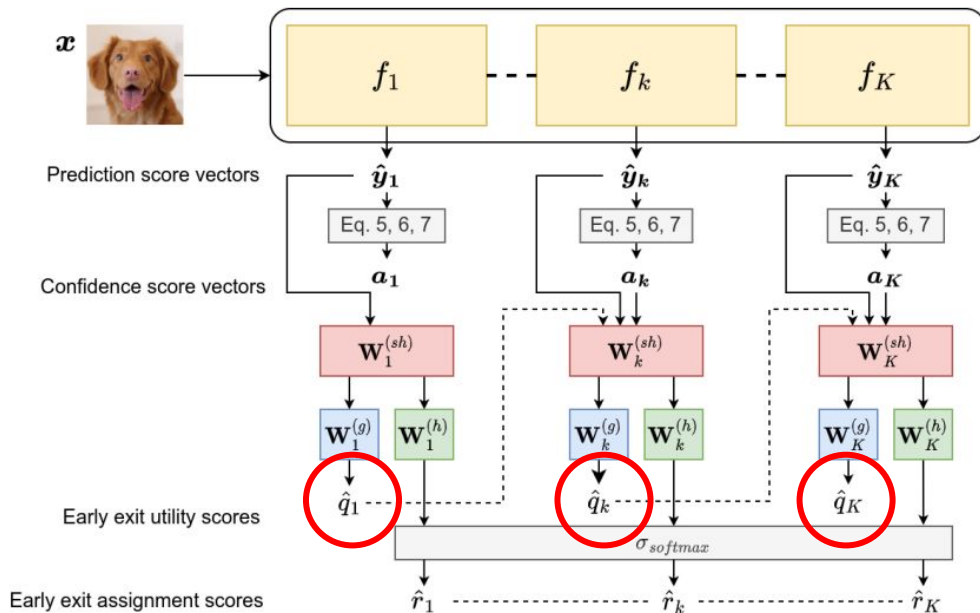


Figure 2: System architecture of EENet.

Architecture

Early exit assignment scores averaged over all samples give us exit distribution.

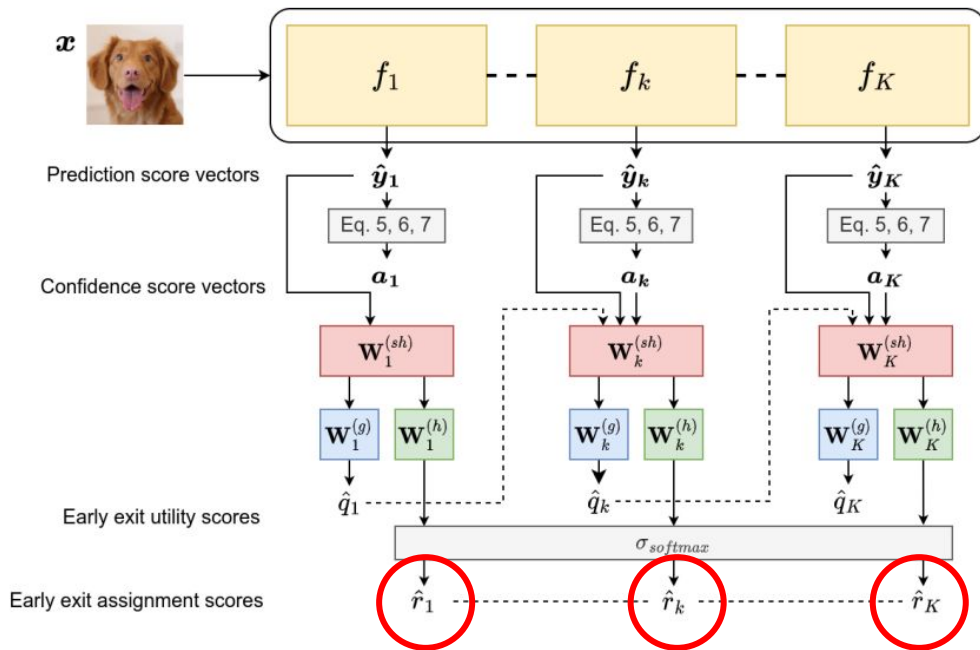


Figure 2: System architecture of EENet.

How to use the results?

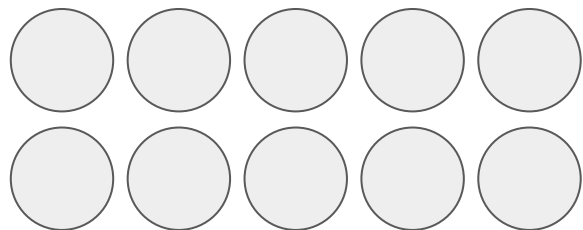
What do we know?

- Exit distribution. It's the mean of the exit distribution scores.
- Samples more likely to be correctly classified thanks to the exit score.

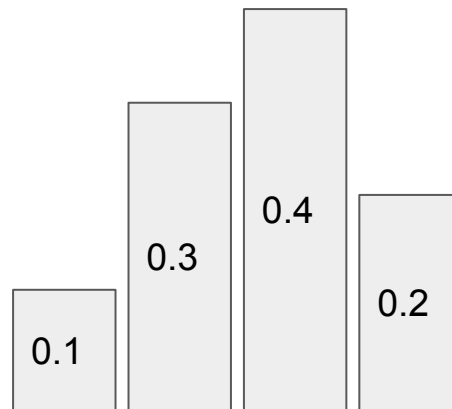
Then let's calculate exit scores thresholds for each exit which tells each exit what exit scores it should tolerate.

These thresholds together with exit scores are enough to decide on exit.

Exit score thresholds example

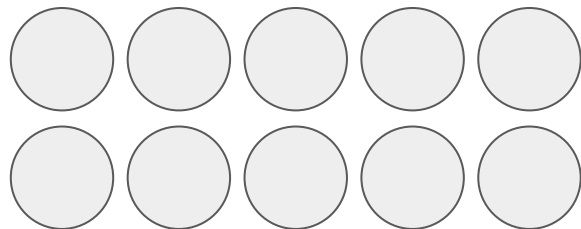


Samples

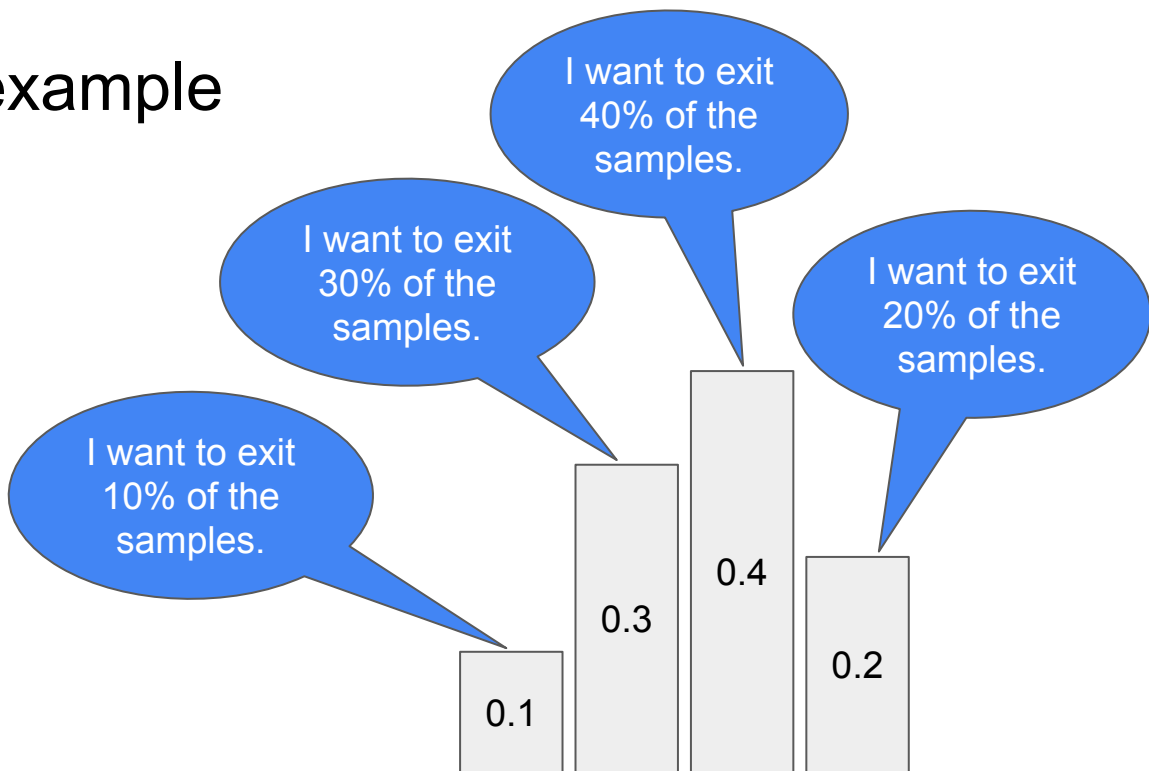


Exits

Exit score thresholds example

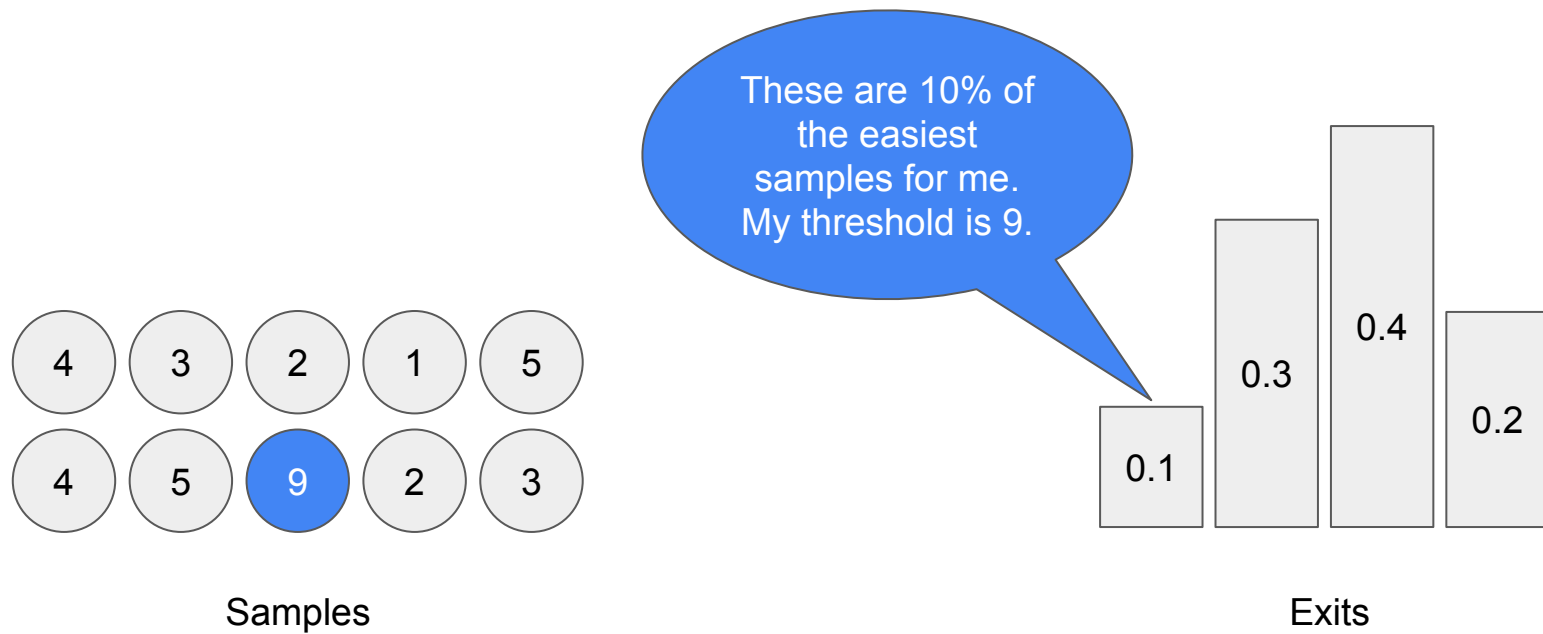


Samples

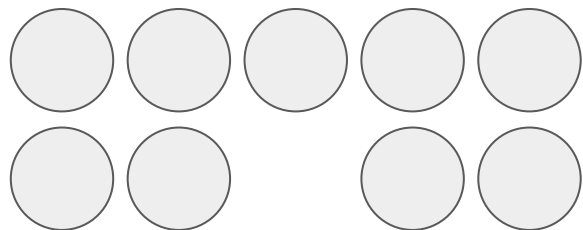


Exits

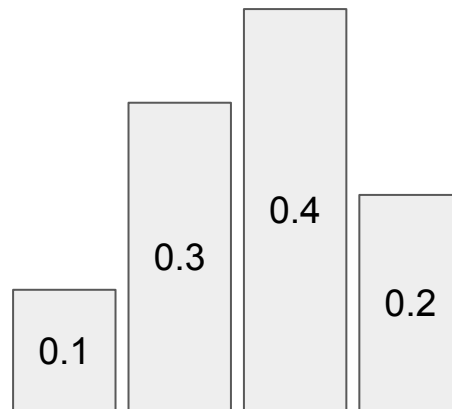
Exit score thresholds example



Exit score thresholds example

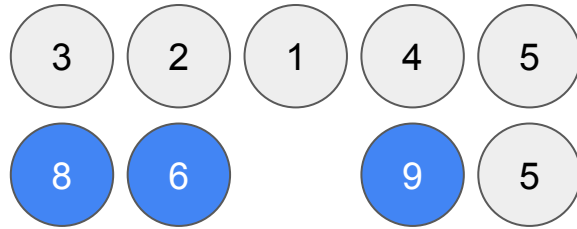


Samples



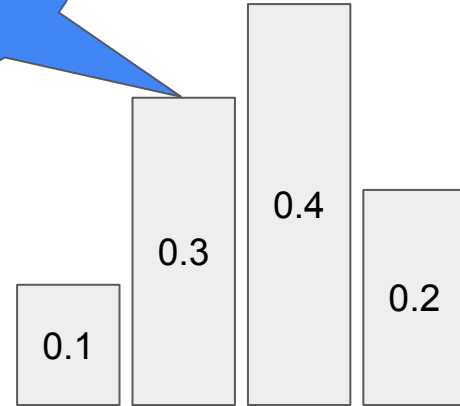
Exits

Exit score thresholds example



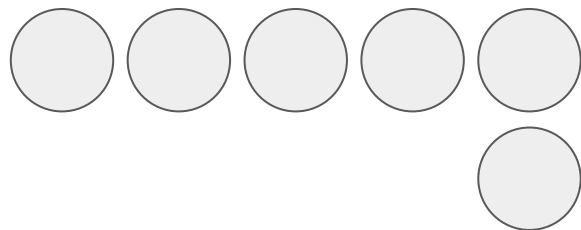
Samples

These are 30% of easiest samples for me. My threshold is 6.

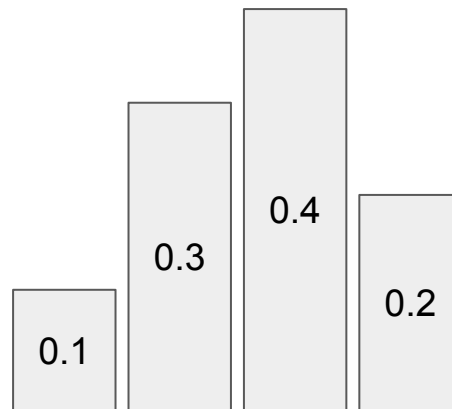


Exits

Exit score thresholds example

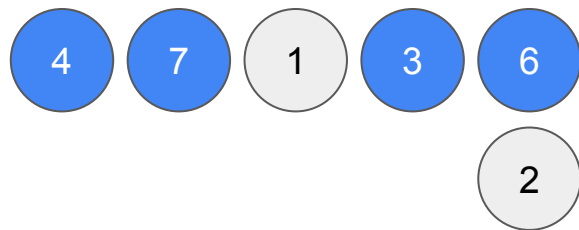


Samples

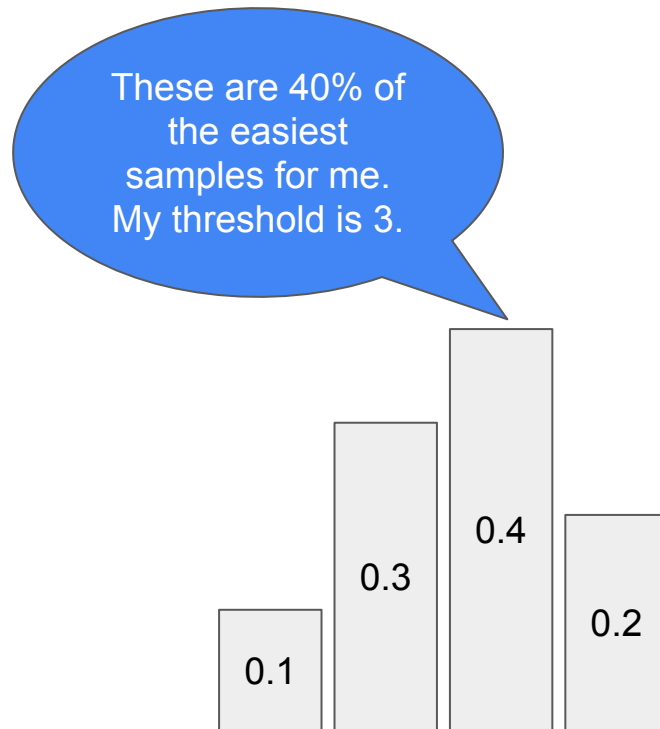


Exits

Exit score thresholds example

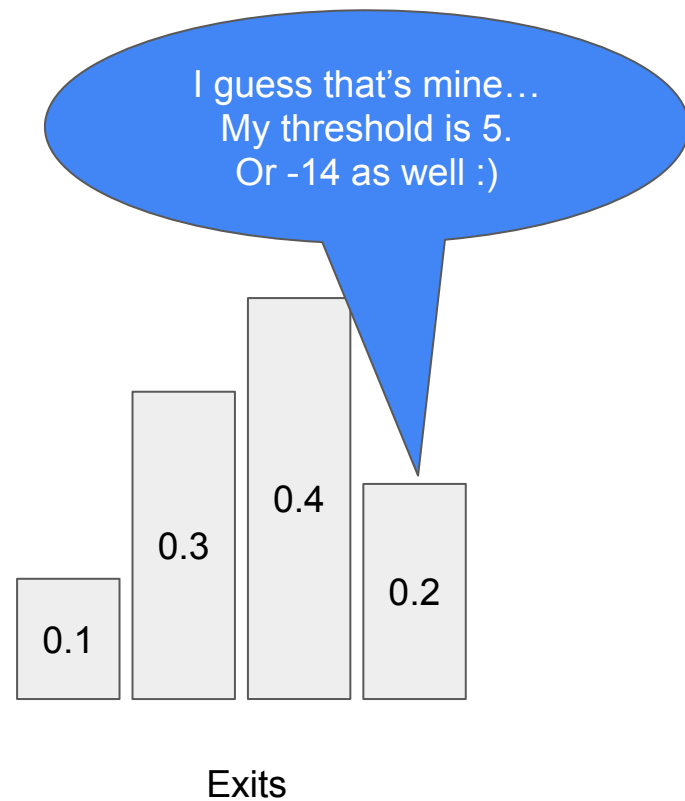
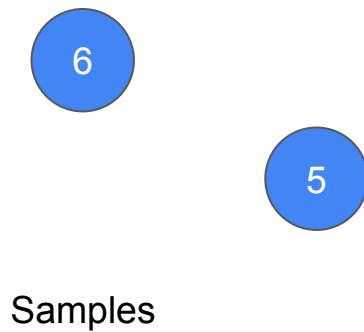


Samples

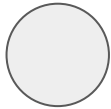


Exits

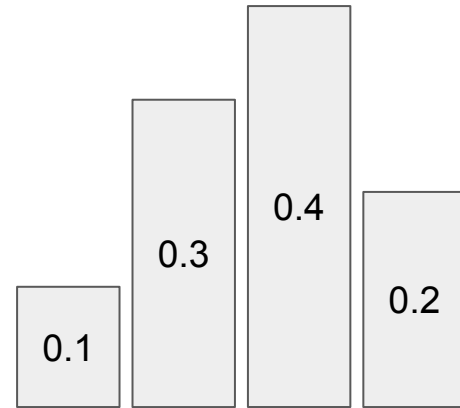
Exit score thresholds example



Exit score thresholds example

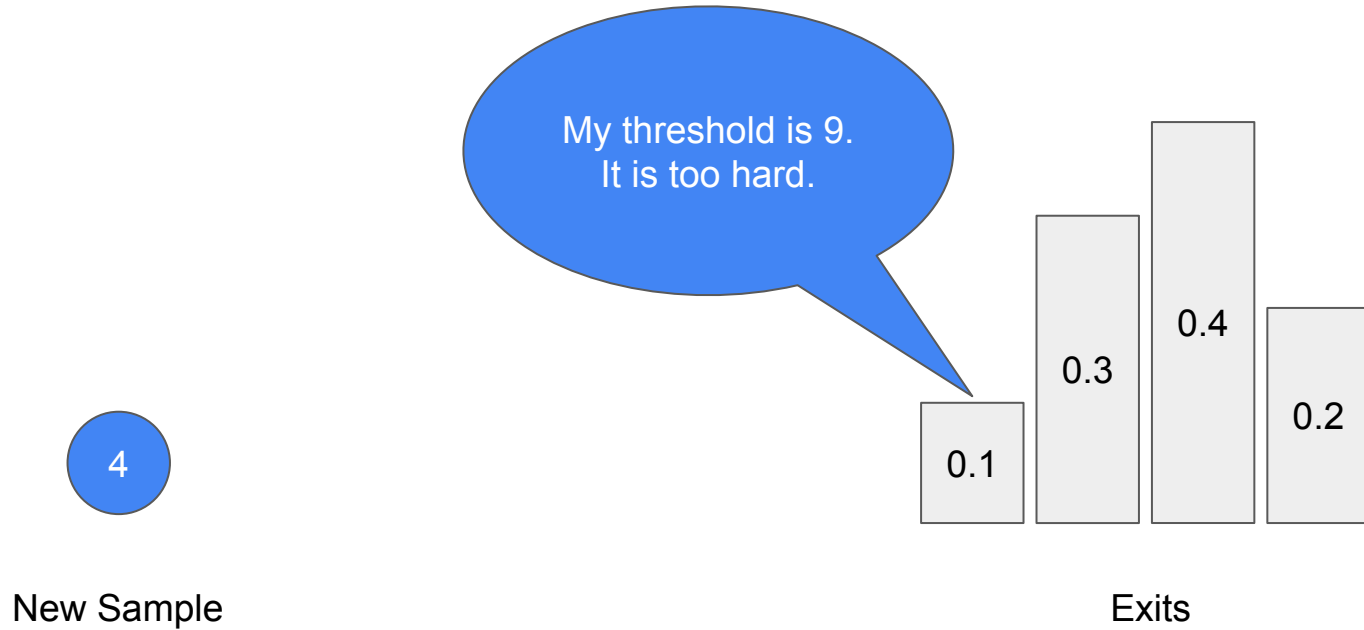


New Sample

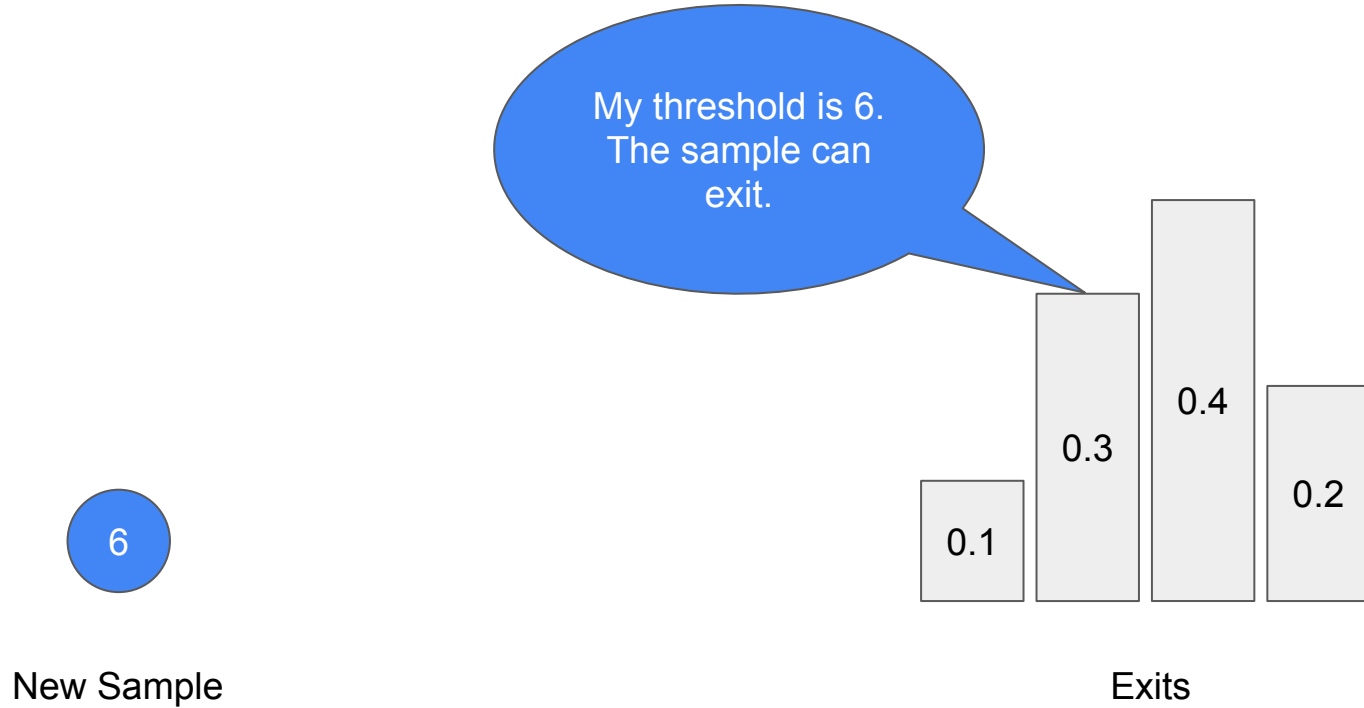


Exits

Exit score thresholds example



Exit score thresholds example

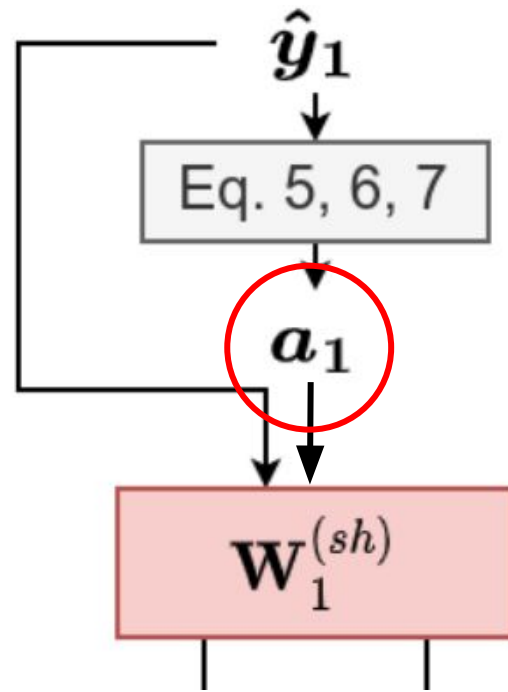


Confidence score vectors

$$a_k^{(max)} = \max_{c \in \mathcal{C}} \hat{y}_{k,c},$$

$$a_k^{(entropy)} = 1 + \frac{\sum_{c'=1}^C \hat{y}_{k,c'} \log \hat{y}_{k,c'}}{\log C},$$

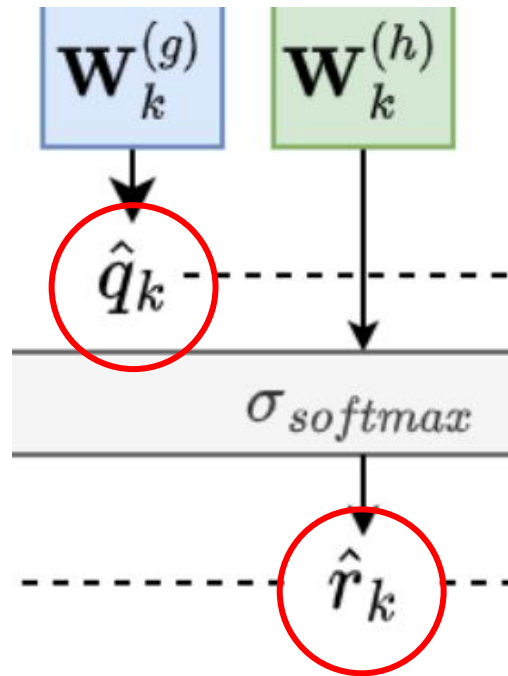
$$a_k^{(vote)} = \frac{1}{k} \max_{c \in \mathcal{C}} \sum_{k'=1}^k \mathbf{1}_{\hat{y}_{k'}=c}.$$



Target variables

During the training, the highlighted values are pulled towards target variables.

$$q_k = \begin{cases} 1 & \text{if } \hat{y}_k = y \\ 0 & \text{if } \hat{y}_k \neq y \end{cases}$$
$$r_k = \begin{cases} \frac{1}{\sum_{k'=1}^K q_{k'}} & \text{if } \hat{y}_k = y \\ \frac{1}{K} & \text{if } \hat{y}_{k'} \neq y \quad \forall k' \in \{1 \dots K\} \\ 0 & \text{otherwise,} \end{cases}$$



Loss function 1

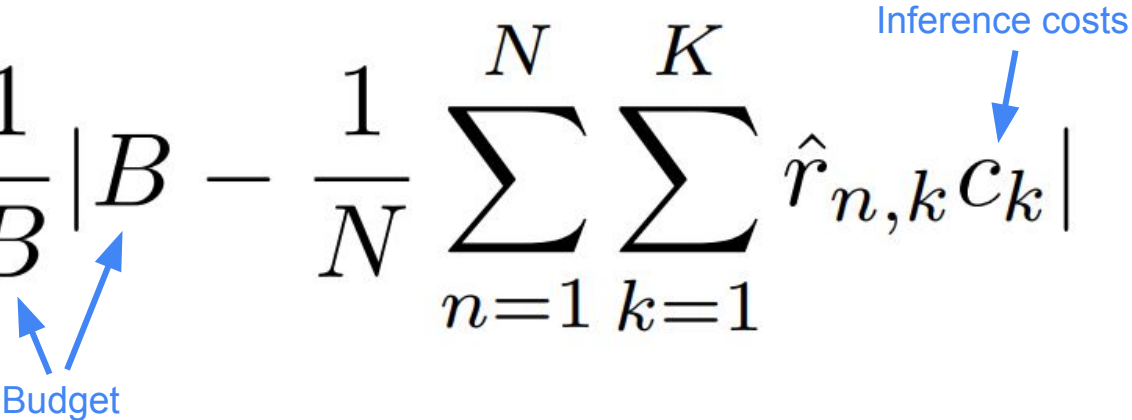
We pull the exit assignment scores towards predefined target.

$$\mathcal{L}_{CE} = -\frac{1}{NK} \sum_{n=1}^N \sum_{k=1}^K \log(\hat{r}_{n,k}) r_{n,k},$$

Loss function 2

The exit assignment can't exceed the budget. Mean exit time must be low enough.

$$\mathcal{L}_{budget} = \frac{1}{B} |B - \frac{1}{N} \sum_{n=1}^N \sum_{k=1}^K \hat{r}_{n,k} c_k|$$



Budget

Inference costs

Loss function 3

Exit scores are trained to predict if the predictions are correct.

$$\mathcal{L}_g = \frac{1}{K} \sum_{n=1}^N \sum_{k=1}^K \ell_g(\hat{q}_{n,k}, q_{n,k}) \text{ such that}$$

$$\ell_g(\hat{q}_k, q_k) = q_k \log(\hat{q}_k) + (1 - q_k) \log(1 - \hat{q}_k)$$

Loss function 3

Exit scores are trained to predict if the predictions are correct.

Especially on samples which should be classified after or at current exit.

$$\mathcal{L}_g = \frac{1}{K} \sum_{n=1}^N \sum_{k=1}^K w_{n,k} \ell_g(\hat{q}_{n,k}, q_{n,k}) \text{ such that}$$

$$\ell_g(\hat{q}_k, q_k) = q_k \log(\hat{q}_k) + (1 - q_k) \log(1 - \hat{q}_k) \text{ and}$$

$$w_{n,k} = \frac{1 - \sum_{k'=1}^{k-1} \hat{r}_{k',n}}{\sum_{n'=1}^N (1 - \sum_{k'=1}^{k-1} \hat{r}_{k',n})},$$

Loss function 3

The more the sample should exit after or at the current exit, the higher value w .


$$\mathcal{L}_g = \frac{1}{K} \sum_{n=1}^N \sum_{k=1}^K \boxed{w_{n,k}} \ell_g(\hat{q}_{n,k}, q_{n,k}) \text{ such that}$$

$$\ell_g(\hat{q}_k, q_k) = q_k \log(\hat{q}_k) + (1 - q_k) \log(1 - \hat{q}_k) \text{ and}$$

$$\boxed{w_{n,k} = \frac{1 - \sum_{k'=1}^{k-1} \hat{r}_{k',n}}{\sum_{n'=1}^N (1 - \sum_{k'=1}^{k-1} \hat{r}_{k',n})}},$$

Loss function 3

The higher value \hat{q} (the higher confidence) on correctly predicted samples.
The lower value \hat{q} (the lower confidence) on wrongly predicted samples.

$$\mathcal{L}_g = \frac{1}{K} \sum_{n=1}^N \sum_{k=1}^K \boxed{w_{n,k}} \boxed{\ell_g(\hat{q}_{n,k}, q_{n,k})} \text{ such that}$$



$$\boxed{\ell_g(\hat{q}_k, q_k) = q_k \log(\hat{q}_k) + (1 - q_k) \log(1 - \hat{q}_k)} \text{ and}$$

$$\boxed{w_{n,k} = \frac{1 - \sum_{k'=1}^{k-1} \hat{r}_{k',n}}{\sum_{n'=1}^N (1 - \sum_{k'=1}^{k-1} \hat{r}_{k',n})}},$$

Loss function 3

Is it really what we want?

The higher value \hat{q} (the higher confidence) on correctly predicted samples.
The lower value \hat{q} (the lower confidence) on wrongly predicted samples.

$$\mathcal{L}_g = \frac{1}{K} \sum_{n=1}^N \sum_{k=1}^K \boxed{w_{n,k}} \boxed{\ell_g(\hat{q}_{n,k}, q_{n,k})} \text{ such that}$$


$$\boxed{\ell_g(\hat{q}_k, q_k) = q_k \log(\hat{q}_k) + (1 - q_k) \log(1 - \hat{q}_k)} \text{ and}$$

$$\boxed{w_{n,k} = \frac{1 - \sum_{k'=1}^{k-1} \hat{r}_{k',n}}{\sum_{n'=1}^N (1 - \sum_{k'=1}^{k-1} \hat{r}_{k',n})}},$$

Results

Dataset, Model and Base Performance	Average Latency Budget per sample	Accuracy (%)			
		BranchyNet	MSDNet	PABEE	EENet
CIFAR-10	3.50 ms	93.76	93.81	93.69	93.84
ResNet56 w/ 3 exits	3.00 ms	92.57	92.79	91.85	92.90
93.90% @ 4.70 ms	2.50 ms	87.55	88.76	84.39	88.90
CIFAR-100	7.50 ms	73.96	74.01	73.68	74.08
DenseNet121 w/ 4 exits	6.75 ms	71.65	71.99	68.10	72.75
75.08% @ 10.20 ms	6.00 ms	68.13	68.70	61.13	70.15
ImageNet	1.50 s	74.10	74.13	74.05	74.18
MSDNet35 w/ 5 exits	1.25 s	72.44	72.70	72.40	72.75
74.60% @ 2.35 s	1.00 s	69.32	69.76	68.13	69.88
SST-2	2.50 s	90.86	91.00	90.75	92.07
BERT w/ 4 exits	2.00 s	87.66	87.71	86.99	91.45
92.36% @ 3.72 s	1.75 s	84.33	84.30	80.99	90.45
AgNews	2.50 s	92.95	92.98	92.57	93.84
BERT w/ 4 exits	2.00 s	85.58	84.93	85.22	93.75
93.98% @ 3.72 s	1.50 s	75.08	73.07	74.67	90.63

Table 1: Image and text classification experiment results in terms of accuracy obtained at different budget levels (average latency per sample).

Results

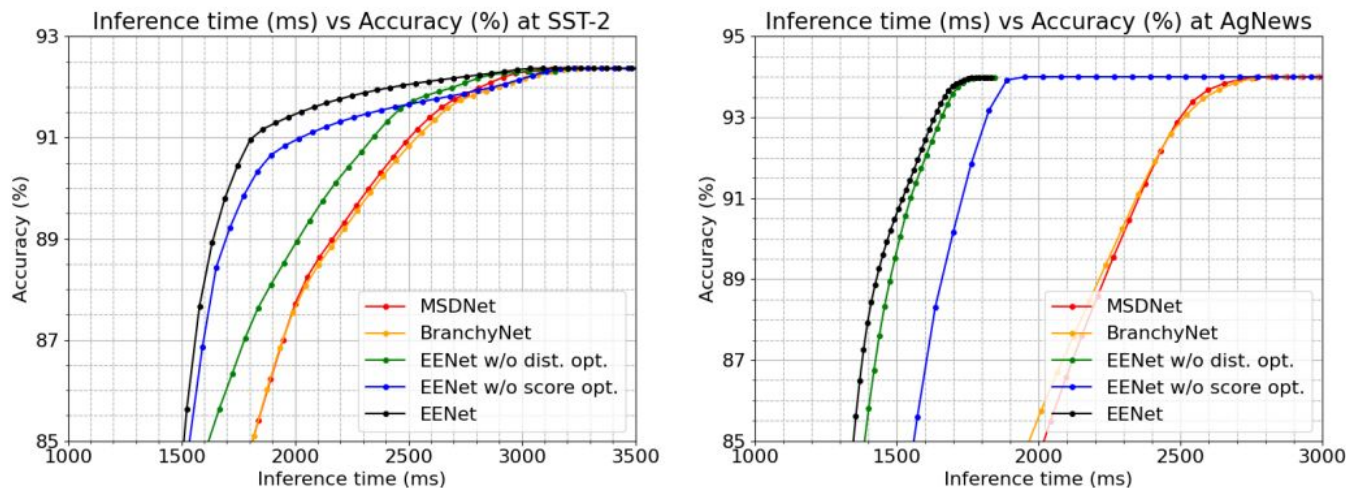


Figure 5: Average latency (ms) vs Accuracy (%) results at SST-2 and AgNews datasets for BranchyNet, MSDNet and EENet variations (without distribution/scoring optimization).

Maybe it is what we want...

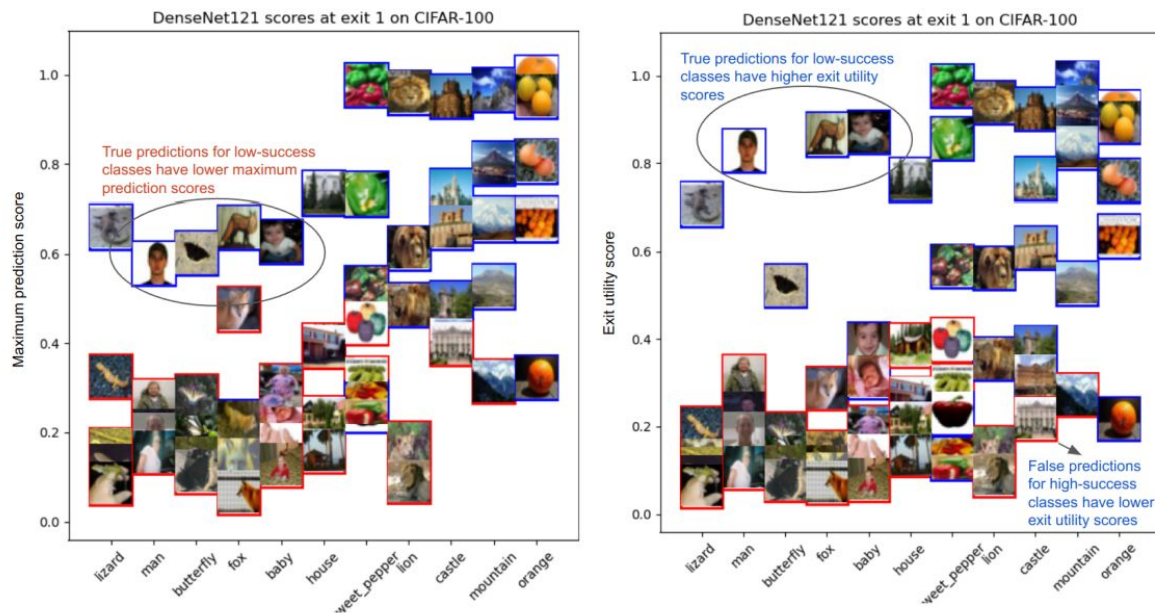


Figure 3: Analysis on the benefit of exit utility scores obtained by EENet, which provides a clearer separation of true and false predictions for all classes, compared to maximum prediction score based confidence, which is popularly used in the literature. Images with blue/red borders are predicted correctly/incorrectly at the first exit of DenseNet121.

Example assignment

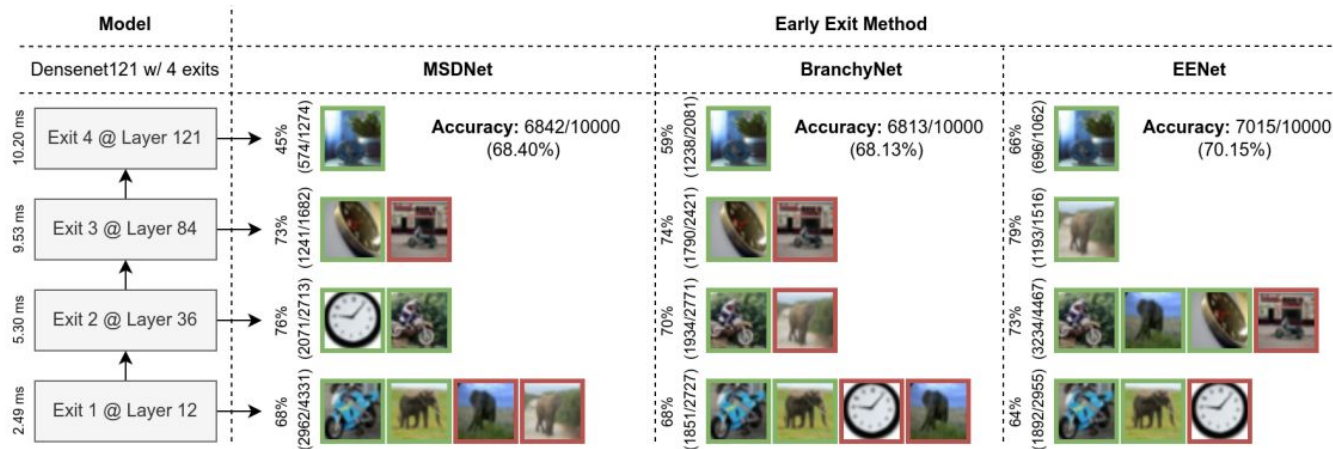


Figure 4: Visual comparison of the early exit approaches on CIFAR-100 test data with DenseNet121 (4 exits) for the average latency budget of 6 ms. We illustrate the randomly selected nine samples from three classes and the exit location that they were assigned. Images with green/red borders are predicted correctly/incorrectly at the corresponding exit. We also report the number of correct predictions and exited samples at each exit. In this case, EENet obtains the performance gain by allowing more samples to exit at the second exit.

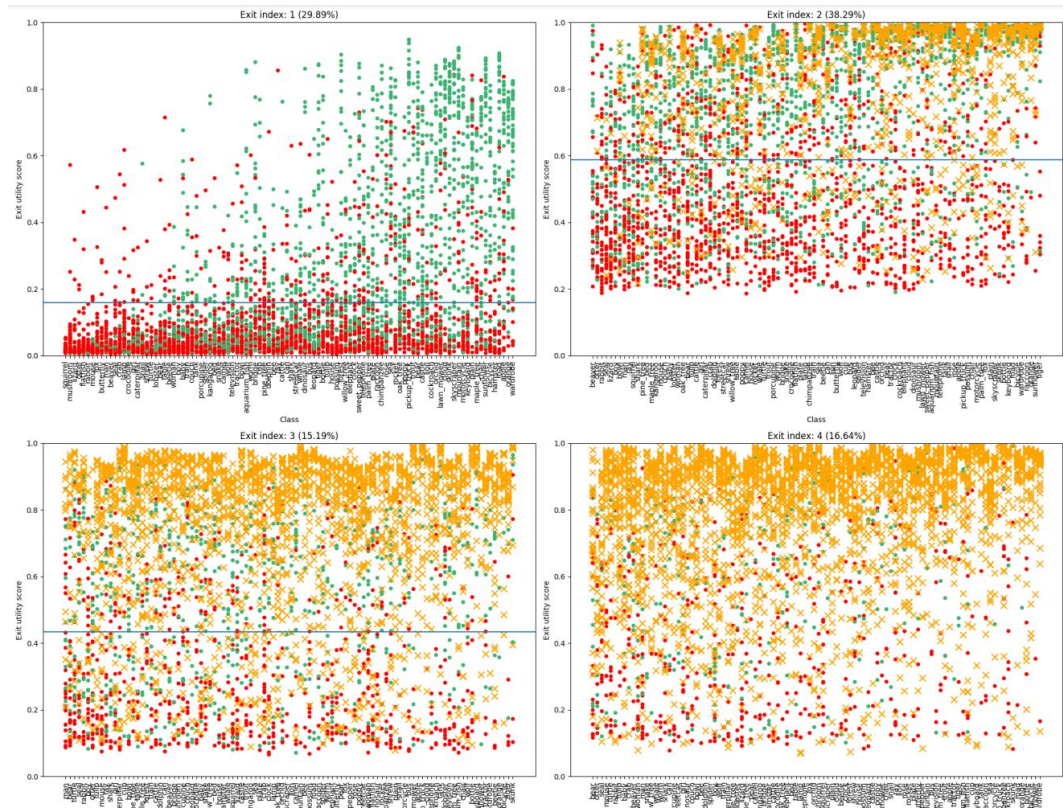
Exit behavior

Green - correct

Red - wrong

Yellow - exited earlier

Line - threshold



Questions