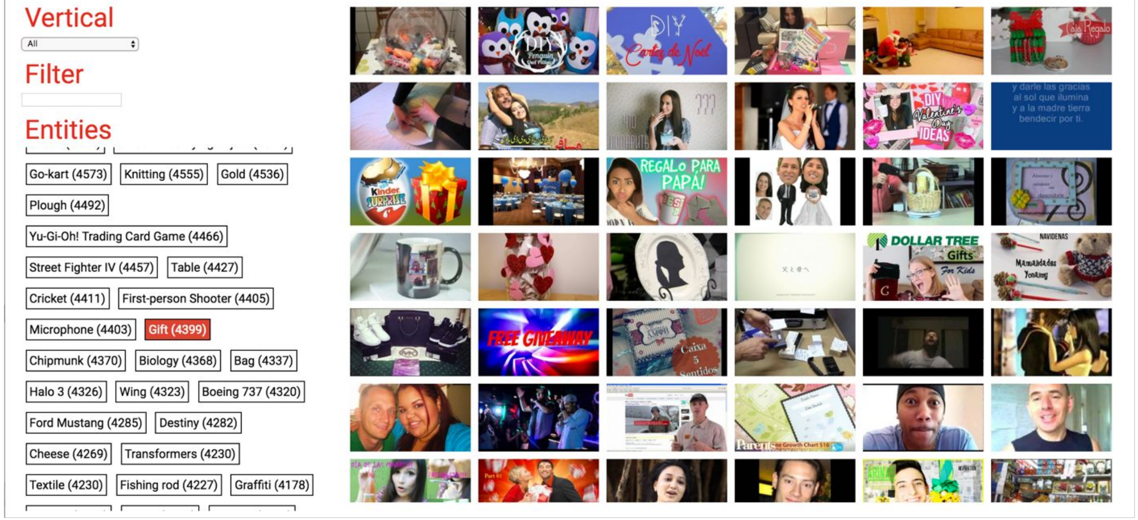


Vertical

All

Filter

Entities

Go-kart (4573) Knitting (4555) Gold (4536)

Plough (4492)

Yu-Gi-Oh! Trading Card Game (4466)

Street Fighter IV (4457) Table (4427)

Cricket (4411) First-person Shooter (4405)

Microphone (4403) Gift (4399)

Chipmunk (4370) Biology (4368) Bag (4337)

Halo 3 (4326) Wing (4323) Boeing 737 (4320)

Ford Mustang (4285) Destiny (4282)

Cheese (4269) Transformers (4230)

Textile (4230) Fishing rod (4227) Graffiti (4178)

Google Cloud Platform

Large-Scale Video Classification with Cloud Computing

Tyler Kohan, Murat Turkeli, Yunsheng Bai

TensorFlow

Introduction

- Kaggle contest: Google Cloud & YouTube-8M Video Understanding Challenge

kaggle

$100,000 prize money

- Hosted by Google / Youtube

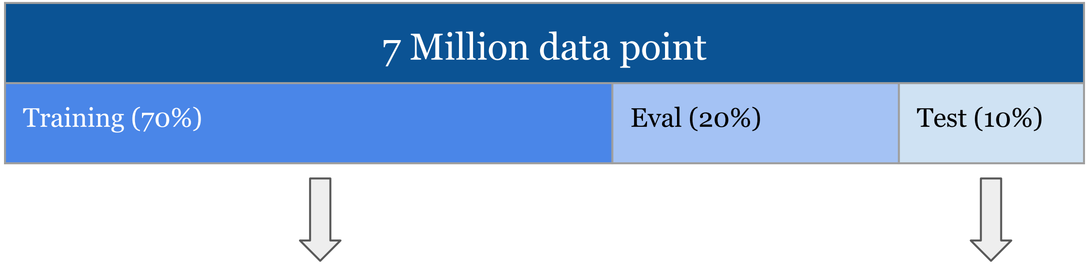
Google

YouTube

- Classify a video into multiple classes

Youtube-8M v2 Video Dataset

- Training dataset: with labels, lots of videos
- Testing dataset: no labels, less videos
- Each video:
  - **rgb** + **audio** features
  - Multiple labels for each video



7 Million data point

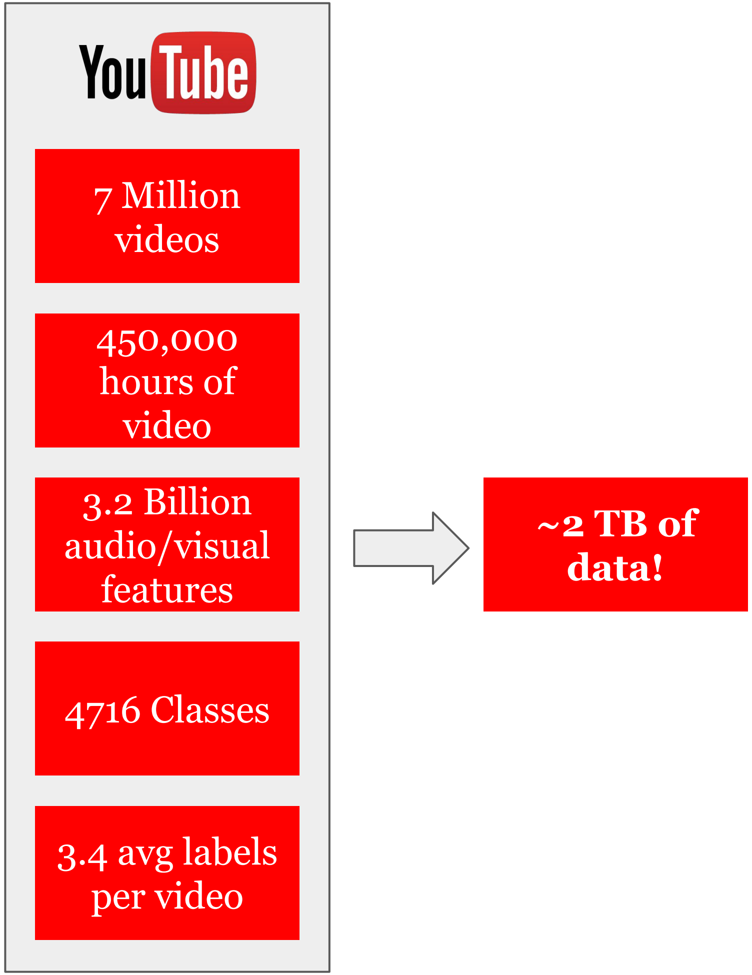
Training (70%)

Eval (20%)

Test (10%)

Train our models

Submission csv file



YouTube

7 Million videos

450,000 hours of video

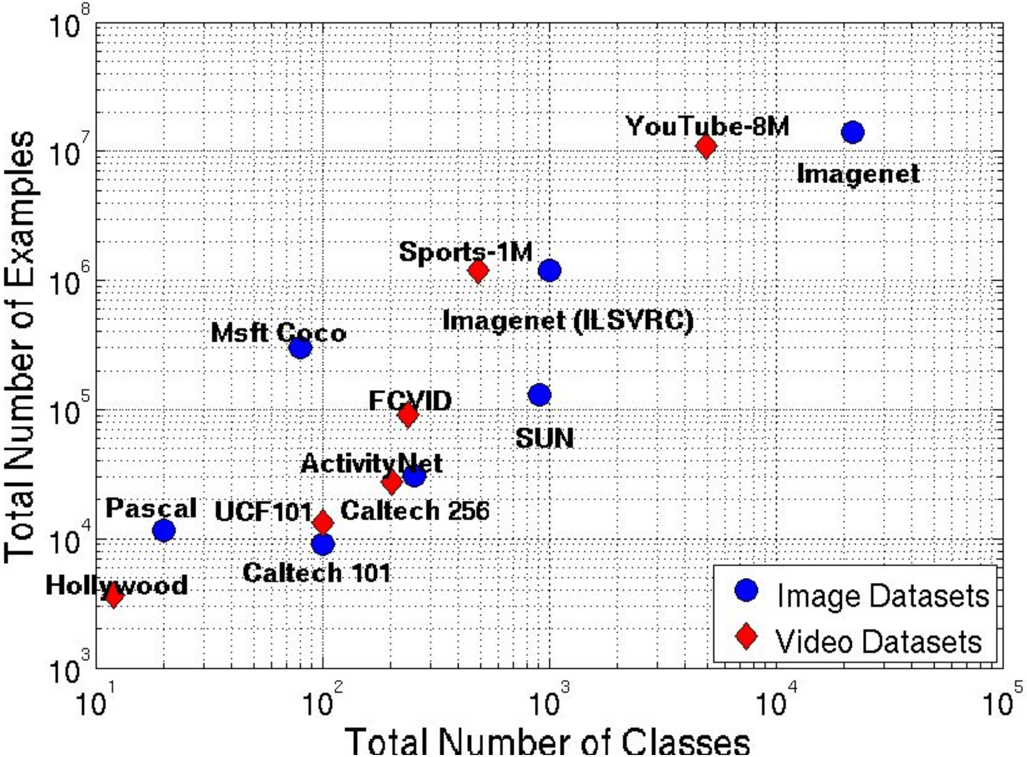
3.2 Billion audio/visual features

4716 Classes

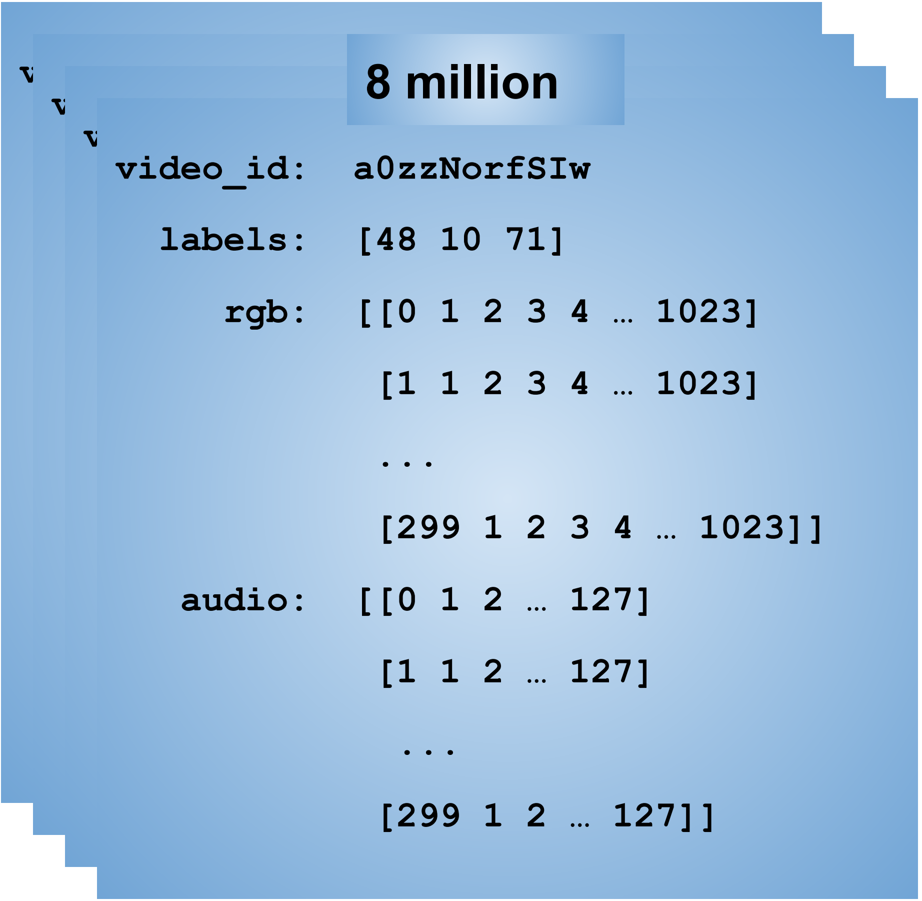
3.4 avg labels per video

~2 TB of data!

Large datasets are good for advances in both image and video understanding tasks.



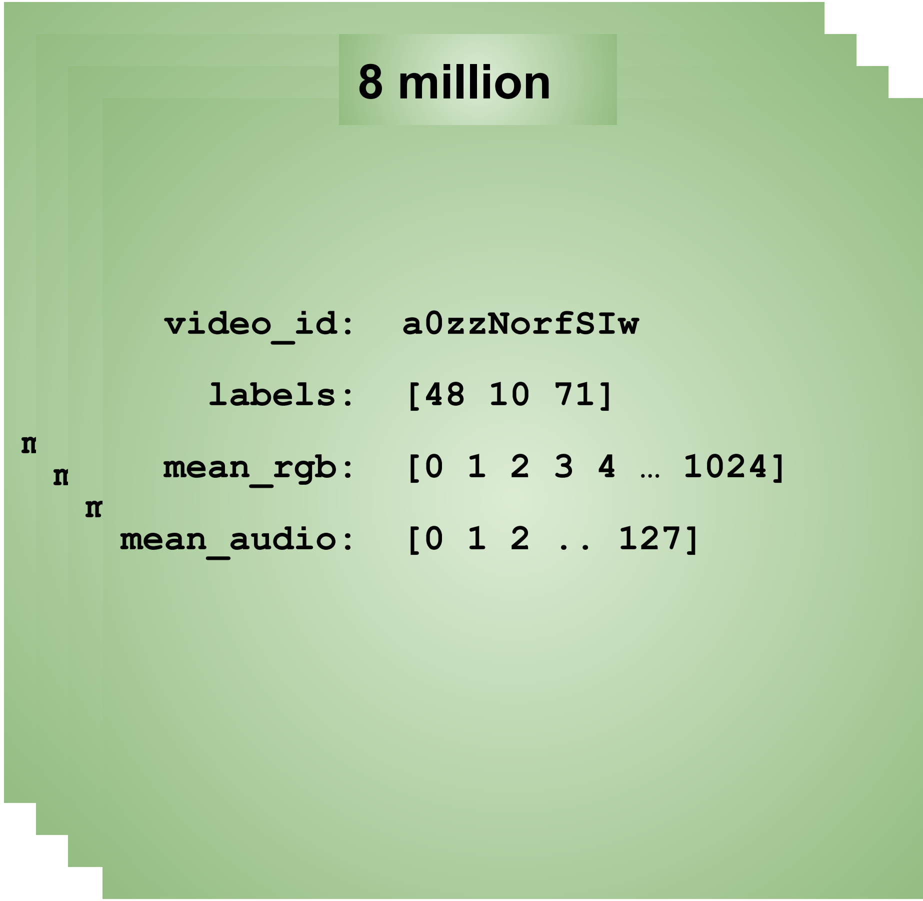
Frame-level features dataset (1.71 TB)



8 million

```
video_id:  a0zzNorfSIw
  labels:  [48 10 71]
     rgb:  [[0 1 2 3 4 … 1023]
            [1 1 2 3 4 … 1023]
            ...
            [299 1 2 3 4 … 1023]]
   audio:  [[0 1 2 … 127]
            [1 1 2 … 127]
             ...
            [299 1 2 … 127]]
```

Video-level features dataset (30 GB)



8 million

```
  video_id:  a0zzNorfSIw
    labels:  [48 10 71]
  mean_rgb:  [0 1 2 3 4 … 1024]
mean_audio:  [0 1 2 .. 127]
```

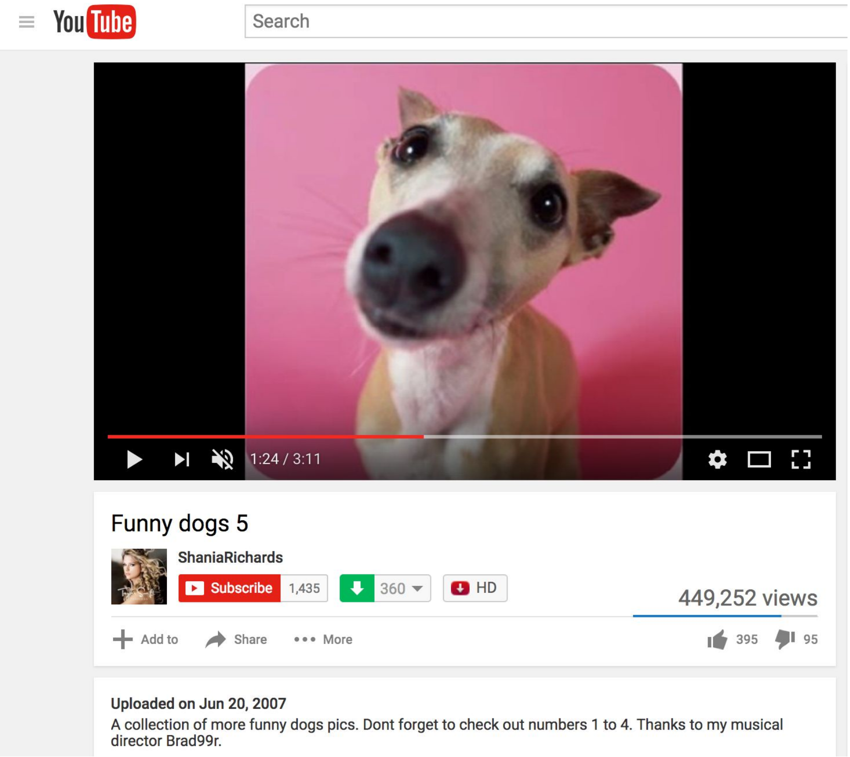
```
video_id:  a0zzNorfSIw
  labels:  [48 10 71]
     rgb:  [[0 1 2 3 4 … 1023]
                      4
             ...
            [299 1 2 3 4 … 1023]]
   audio:  [[0 1 2 … 127]
            [1 1 2 … 127]
              ...
            [299 1 2 … 127]]
```

Pet

Animal

Dog

https://www.youtube.com/watch?v=a0zzNorfSIw



Funny dogs 5

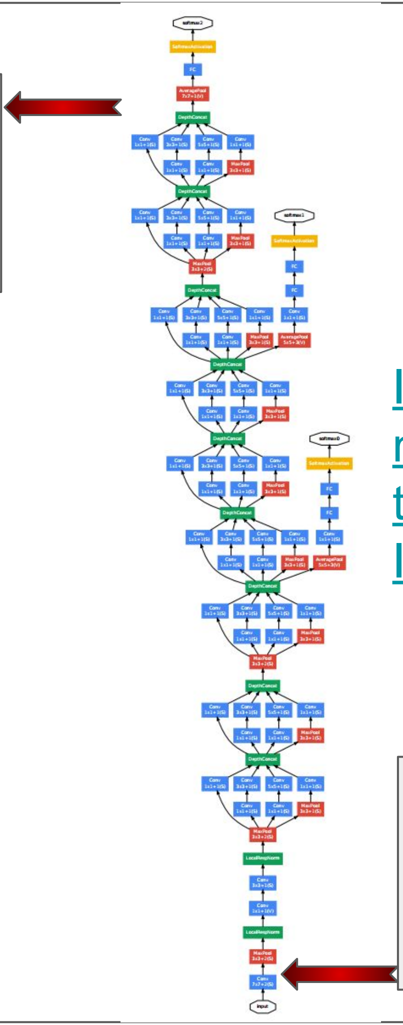
ShaniaRichards

449,252 views

Uploaded on Jun 20, 2007

A collection of more funny dogs pics. Dont forget to check out numbers 1 to 4. Thanks to my musical director Brad99r.

```
video_id:  a0zzNorfSIw
  labels:  [48 10 71]
     rgb:  [[0 1 2 3 4 … 1023]
            [1 1 2 3 4 … 1023]
            ...
            [299 1 2 3 4 … 1023]]
   audio:  [[0 1 2 … 127]
            [1 1 2 … 127]
             ...
            [299 1 2 … 127]]
```



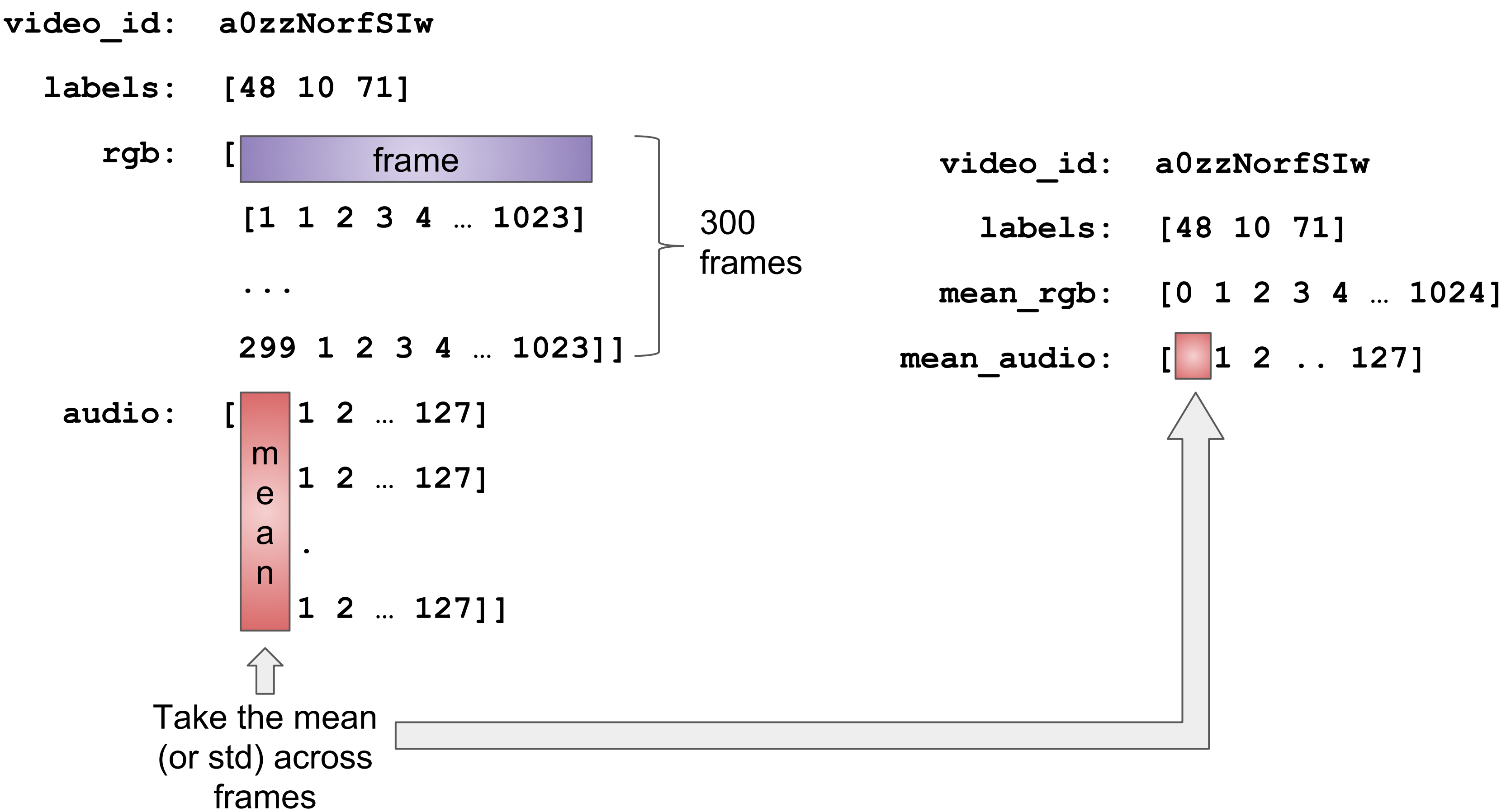
ReLU activation of the last hidden layer

Inception network trained on ImageNet

Raw pixels

Frame-level features dataset

Video-level features dataset



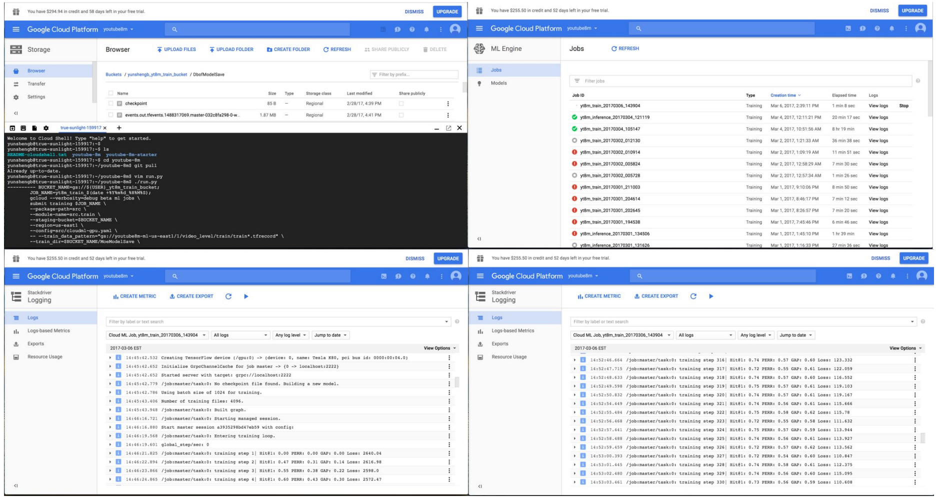
```
video_id:  a0zzNorfSIw

  labels:  [48 10 71]

     rgb:  [ frame
            [1 1 2 3 4 … 1023]

            ...

            299 1 2 3 4 … 1023]]

   audio:  [ 1 2 … 127]

             1 2 … 127]

             .

             1 2 … 127]]
```

Take the mean (or std) across frames

```
  video_id:  a0zzNorfSIw

    labels:  [48 10 71]

  mean_rgb:  [0 1 2 3 4 … 1024]

mean_audio:  [ 1 2 .. 127]
```

TensorFlow and Google Cloud ML

- **TensorFlow:** open source software library that makes it easy for us to perform complex machine learning concepts with limited knowledge
- **Google Cloud Machine Learning:** neural-net ML service that includes a platform to create our own training models



Buckets / yunshengb_yt8m_train_bucket / DbofModelSave

Name	Size	Type	Storage class	Last modified	Share publicly
checkpoint	85 B	–	Regional	2/28/17, 4:39 PM	□
events.out.tfevents.1488317069.master-032c8fa298-0-w...	1.87 MB	–	Regional	2/28/17, 4:41 PM	□

```
Welcome to Cloud Shell! Type "help" to get started.
yunshengb@true-sunlight-159917:~$
yunshengb@true-sunlight-159917:~$ ls
README-cloudshell.txt  youtube-8m  youtube-8m-starter
yunshengb@true-sunlight-159917:~$ cd youtube-8m
yunshengb@true-sunlight-159917:~/youtube-8m$ git pull
Already up-to-date.
yunshengb@true-sunlight-159917:~/youtube-8m$ vim run.py
yunshengb@true-sunlight-159917:~/youtube-8m$ ./run.py
---------- BUCKET_NAME=gs://${USER}_yt8m_train_bucket;
        JOB_NAME=yt8m_train_$(date +%Y%m%d_%H%M%S);
        gcloud --verbosity=debug beta ml jobs \
        submit training $JOB_NAME \
        --package-path=src \
        --module-name=src.train \
        --staging-bucket=$BUCKET_NAME \
        --region=us-east1 \
        --config=src/cloudml-gpu.yaml \
        -- --train_data_pattern="gs://youtube8m-ml-us-east1/1/video_level/train/train*.tfrecord" \
        --train_dir=$BUCKET_NAME/MoeModelSave \
```

Job ID	Type	Creation time	Elapsed time	Logs	
yt8m_train_20170306_143904	Training	Mar 6, 2017, 2:39:11 PM	1 min 8 sec	View logs	Stop
✓ yt8m_inference_20170304_121119	Training	Mar 4, 2017, 12:11:21 PM	20 min 17 sec	View logs	
✓ yt8m_train_20170304_105147	Training	Mar 4, 2017, 10:51:56 AM	8 hr 19 min	View logs	
○ yt8m_train_20170302_012130	Training	Mar 2, 2017, 1:21:33 AM	36 min 38 sec	View logs	
! yt8m_train_20170302_010914	Training	Mar 2, 2017, 1:09:19 AM	11 min 51 sec	View logs	
! yt8m_train_20170302_005824	Training	Mar 2, 2017, 12:58:29 AM	7 min 30 sec	View logs	
○ yt8m_train_20170302_005728	Training	Mar 2, 2017, 12:57:34 AM	1 min 26 sec	View logs	
! yt8m_train_20170301_211003	Training	Mar 1, 2017, 9:10:06 PM	8 min 50 sec	View logs	
! yt8m_train_20170301_204614	Training	Mar 1, 2017, 8:46:17 PM	7 min 12 sec	View logs	
! yt8m_train_20170301_202645	Training	Mar 1, 2017, 8:26:57 PM	7 min 20 sec	View logs	
! yt8m_train_20170301_194538	Training	Mar 1, 2017, 7:45:46 PM	6 min 46 sec	View logs	
! yt8m_inference_20170301_134506	Training	Mar 1, 2017, 1:45:10 PM	1 hr 39 min	View logs	
○ yt8m_inference_20170301_131626	Training	Mar 1, 2017, 1:16:33 PM	27 min 36 sec	View logs	

```
2017-03-06 EST
14:45:42.532 Creating TensorFlow device (/gpu:0) -> (device: 0, name: Tesla K80, pci bus id: 0000:00:04.0)
14:45:42.652 Initialize GrpcChannelCache for job master -> {0 -> localhost:2222}
14:45:42.652 Started server with target: grpc://localhost:2222
14:45:42.779 /job:master/task:0: No checkpoint file found. Building a new model.
14:45:42.786 Using batch size of 1024 for training.
14:45:43.406 Number of training files: 4096.
14:45:43.948 /job:master/task:0: Built graph.
14:46:16.721 /job:master/task:0: Starting managed session.
14:46:16.880 Start master session a3935298bd47eb59 with config:
14:46:19.568 /job:master/task:0: Entering training loop.
14:46:19.601 global_step/sec: 0
14:46:21.825 /job:master/task:0: training step 1| Hit@1: 0.00 PERR: 0.00 GAP: 0.00 Loss: 2640.04
14:46:22.894 /job:master/task:0: training step 2| Hit@1: 0.47 PERR: 0.31 GAP: 0.14 Loss: 2616.98
14:46:23.866 /job:master/task:0: training step 3| Hit@1: 0.55 PERR: 0.38 GAP: 0.22 Loss: 2598.0
14:46:24.865 /job:master/task:0: training step 4| Hit@1: 0.60 PERR: 0.43 GAP: 0.30 Loss: 2572.47
```

```
2017-03-06 EST
14:52:46.664 /job:master/task:0: training step 316| Hit@1: 0.74 PERR: 0.57 GAP: 0.60 Loss: 123.332
14:52:47.715 /job:master/task:0: training step 317| Hit@1: 0.72 PERR: 0.55 GAP: 0.61 Loss: 122.059
14:52:48.633 /job:master/task:0: training step 318| Hit@1: 0.74 PERR: 0.57 GAP: 0.60 Loss: 116.552
14:52:49.598 /job:master/task:0: training step 319| Hit@1: 0.75 PERR: 0.57 GAP: 0.61 Loss: 119.103
14:52:50.832 /job:master/task:0: training step 320| Hit@1: 0.74 PERR: 0.57 GAP: 0.61 Loss: 119.167
14:52:54.449 /job:master/task:0: training step 321| Hit@1: 0.74 PERR: 0.56 GAP: 0.61 Loss: 115.666
14:52:55.484 /job:master/task:0: training step 322| Hit@1: 0.75 PERR: 0.58 GAP: 0.62 Loss: 115.78
14:52:56.468 /job:master/task:0: training step 323| Hit@1: 0.72 PERR: 0.55 GAP: 0.58 Loss: 111.632
14:52:57.441 /job:master/task:0: training step 324| Hit@1: 0.75 PERR: 0.57 GAP: 0.59 Loss: 113.944
14:52:58.488 /job:master/task:0: training step 325| Hit@1: 0.74 PERR: 0.56 GAP: 0.61 Loss: 113.927
14:52:59.459 /job:master/task:0: training step 326| Hit@1: 0.72 PERR: 0.57 GAP: 0.62 Loss: 113.562
14:53:00.393 /job:master/task:0: training step 327| Hit@1: 0.72 PERR: 0.54 GAP: 0.60 Loss: 110.847
14:53:01.445 /job:master/task:0: training step 328| Hit@1: 0.74 PERR: 0.58 GAP: 0.61 Loss: 112.375
14:53:02.480 /job:master/task:0: training step 329| Hit@1: 0.74 PERR: 0.56 GAP: 0.60 Loss: 115.095
14:53:03.461 /job:master/task:0: training step 330| Hit@1: 0.73 PERR: 0.56 GAP: 0.59 Loss: 110.608
```

TensorFlow

python

Google Cloud Platform

Used over $2500 of Google Cloud ML credit!

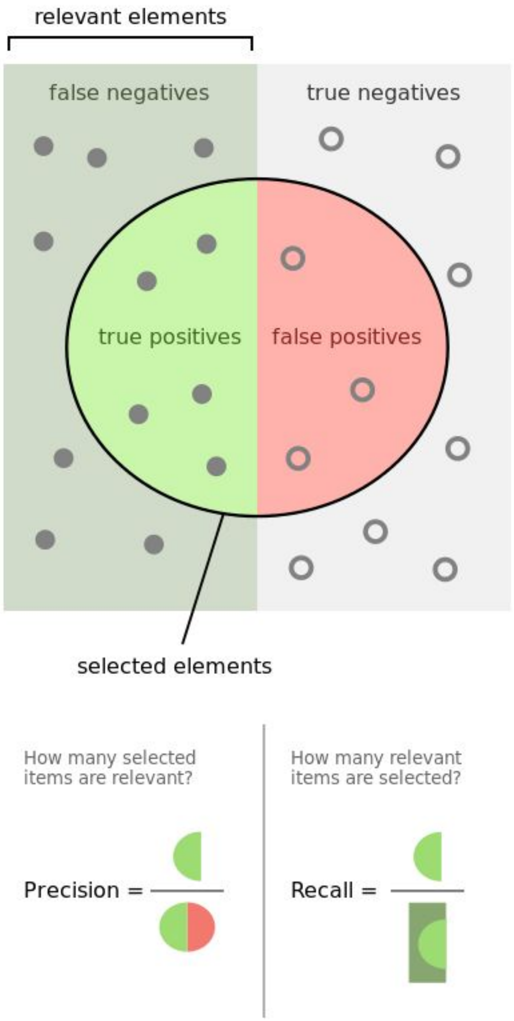
Evaluation

Video ID	Label Confidence Pairs
100011194	1 0.983786 4 0.900343 297 0.891204 2292 0.792589 933 0.688224 ...
100635497	92 0.716859 1 0.714576 926 0.422048 202 0.387686 4 0.254472 ...
...	...

submission.csv

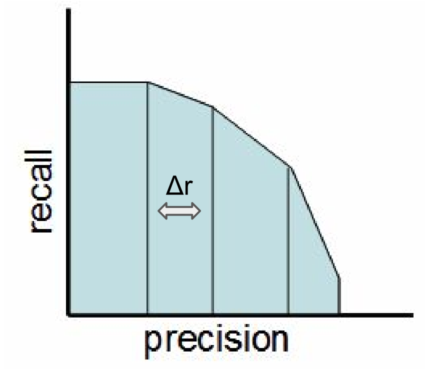
$$GAP = \sum_{i=1}^{N} p(i)\Delta r(i)$$

p(i) is precision, r(i) is the recall, and N is the number of videos



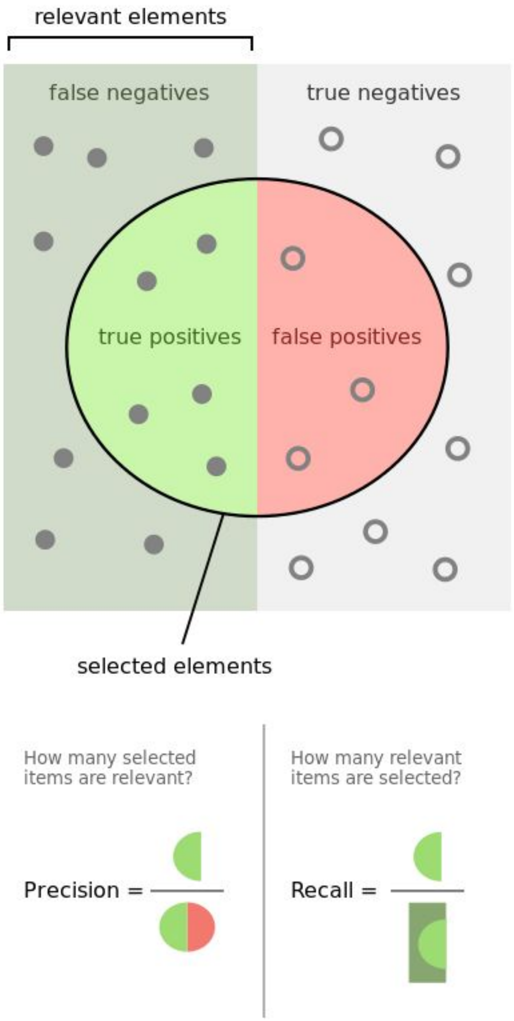
Global Average Precision

- Based on **precision** and **recall**
- Precision: out of the the labels we predicted, how many are correct
- Recall: out of all the actual labels, how many did we find
- Consider only up to 20 label/confidence pair per video



$$GAP = \sum_{i=1}^{N} p(i)\Delta r(i)$$

Corresponds to the **area** under the **curve**

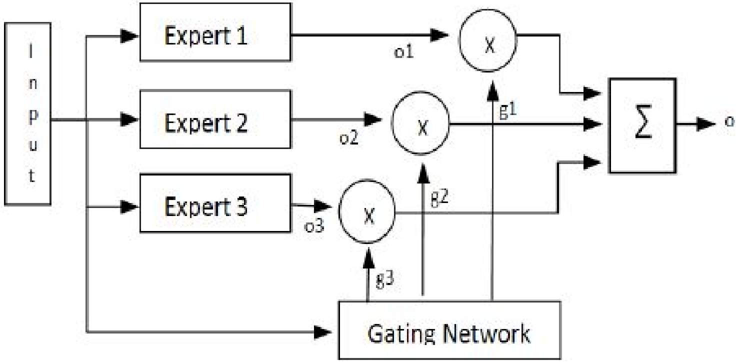


Methods

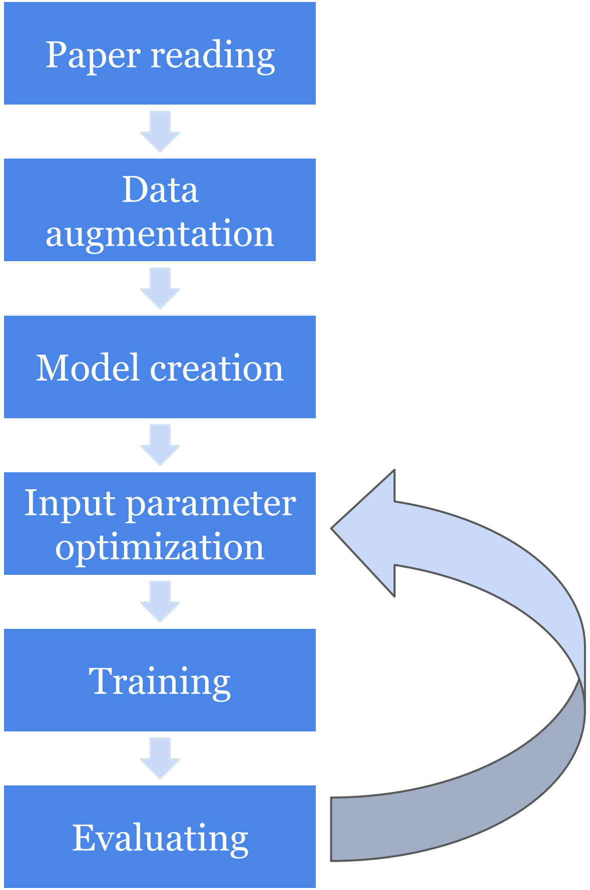
Models we identified as promising

CNN	MOE	LSTM	Adaboost
~60 %	~80%	~78%	?

Mixture of Experts (MOE)



Model development cycle



Paper reading

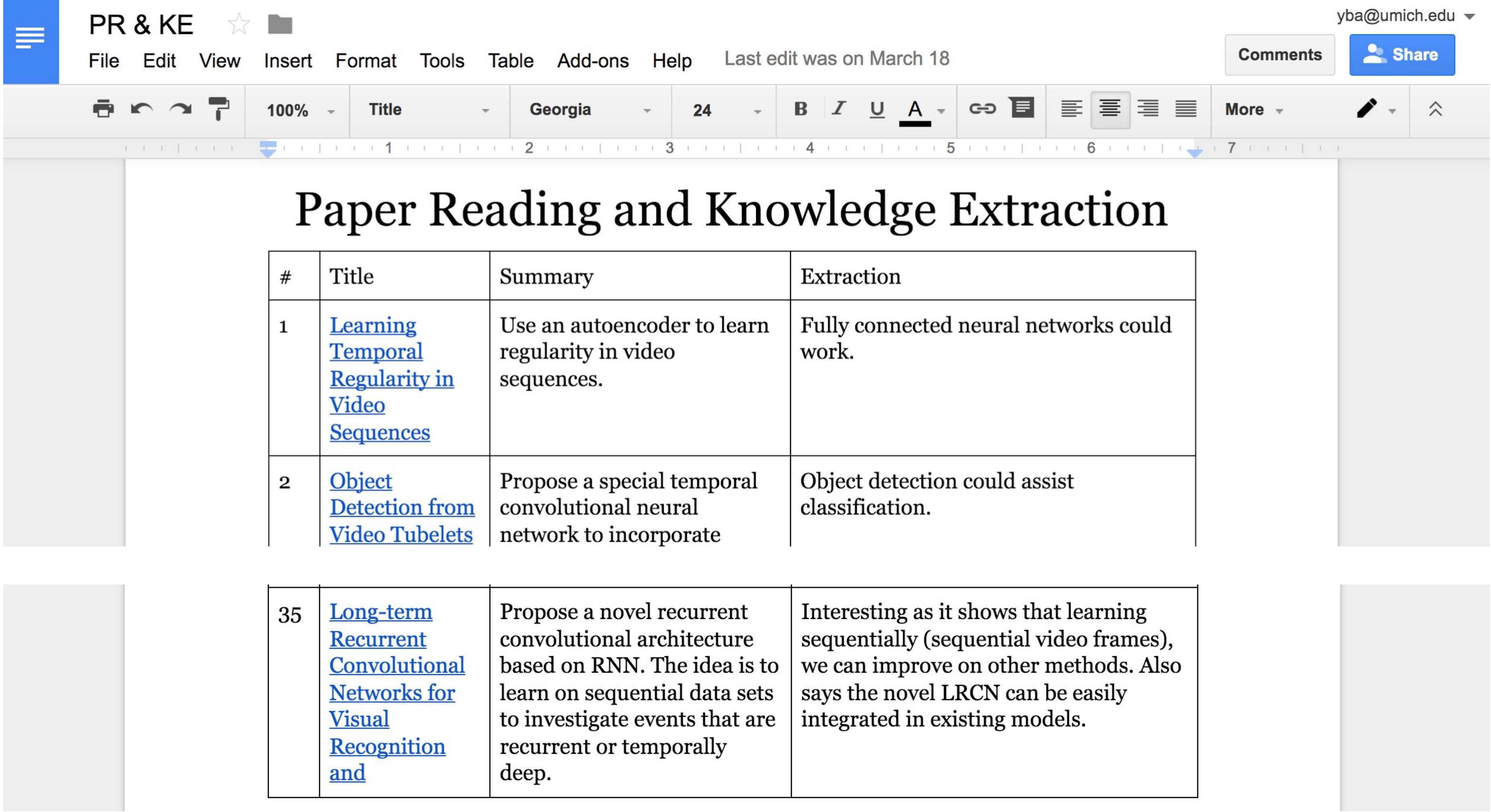
Data augmentation

Model creation

Input parameter optimization

Training

Evaluating

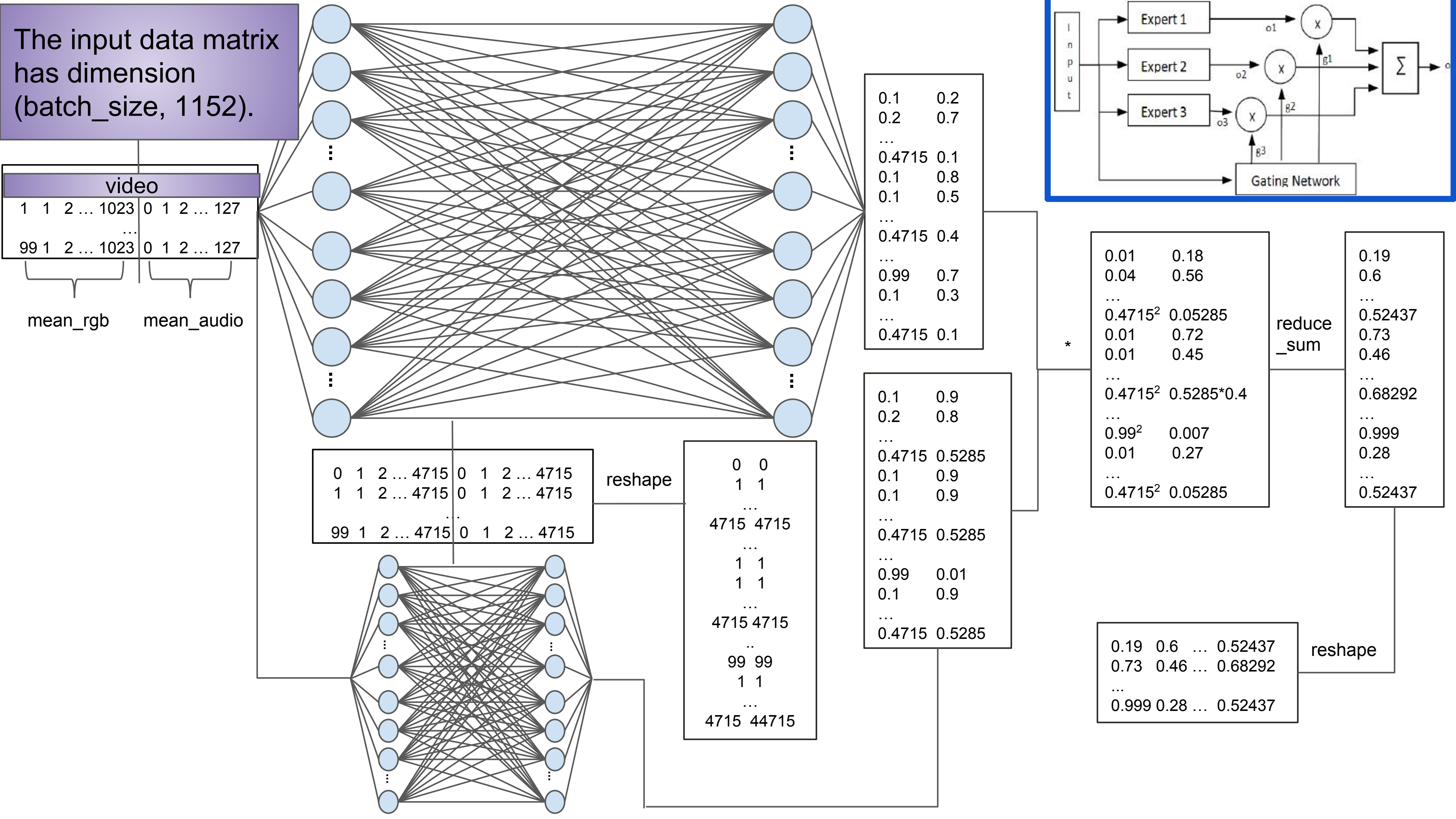


Paper Reading and Knowledge Extraction

#	Title	Summary	Extraction
1	Learning Temporal Regularity in Video Sequences	Use an autoencoder to learn regularity in video sequences.	Fully connected neural networks could work.
2	Object Detection from Video Tubelets	Propose a special temporal convolutional neural network to incorporate	Object detection could assist classification.
35	Long-term Recurrent Convolutional Networks for Visual Recognition and	Propose a novel recurrent convolutional architecture based on RNN. The idea is to learn on sequential data sets to investigate events that are recurrent or temporally deep.	Interesting as it shows that learning sequentially (sequential video frames), we can improve on other methods. Also says the novel LRCN can be easily integrated in existing models.

https://docs.google.com/document/d/1zYSWcDrX38v2glNRXyKmdLDINnkPCHrp01gyqaet2fs/edit?usp=sharing

The input data matrix has dimension (batch_size, 1152).



1	1	2	…	1023	0	1	2	…	127
				…					
99	1	2	…	1023	0	1	2	…	127

0	1	2	…	4715	0	1	2	…	4715
1	1	2	…	4715	0	1	2	…	4715
				…					
99	1	2	…	4715	0	1	2	…	4715

0	0
1	1
…	
4715	4715
…	
1	1
1	1
…	
4715	4715
..	
99	99
1	1
…	
4715	44715

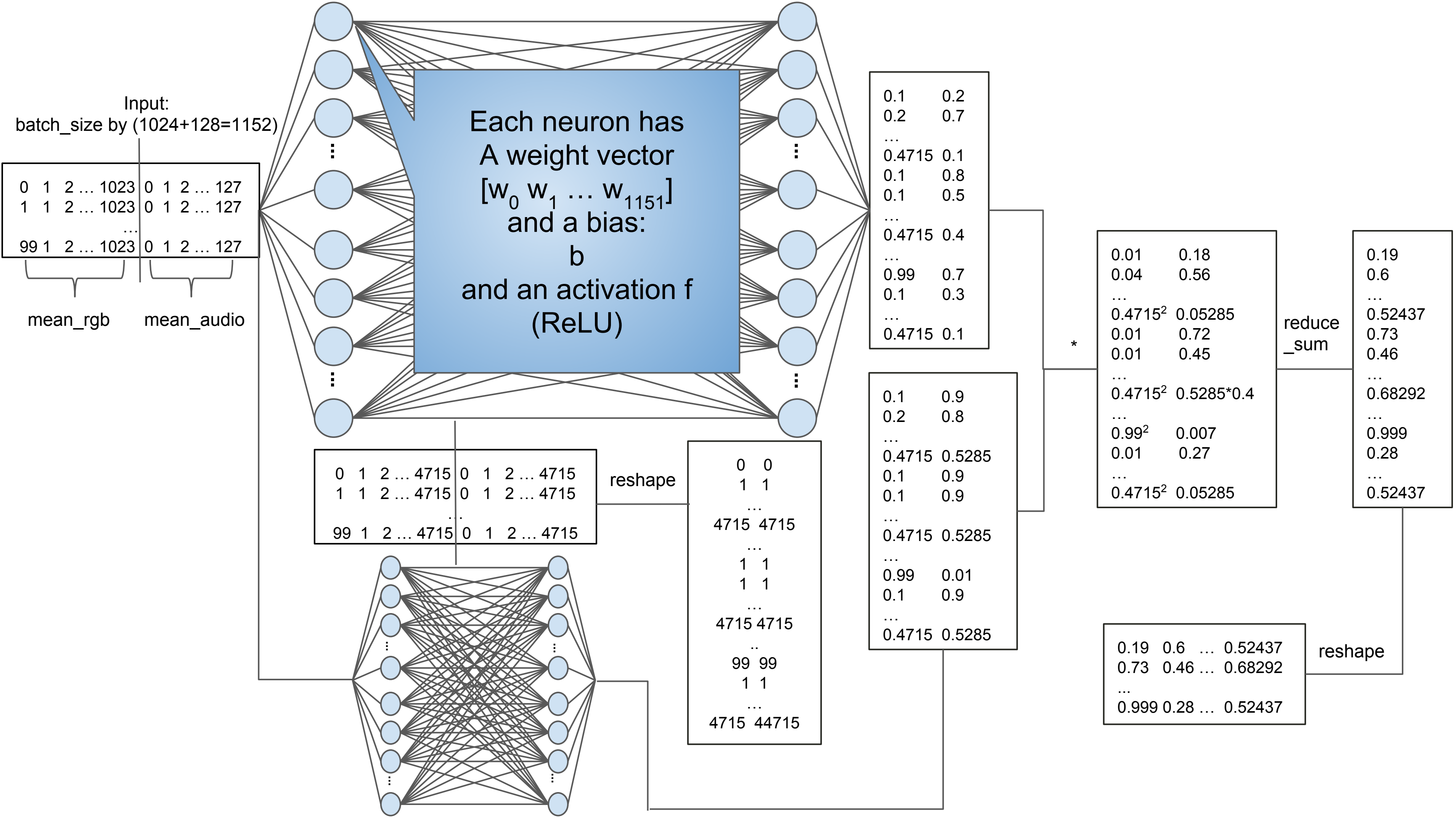
0.1	0.2
0.2	0.7
…	
0.4715	0.1
0.1	0.8
0.1	0.5
…	
0.4715	0.4
…	
0.99	0.7
0.1	0.3
…	
0.4715	0.1

0.1	0.9
0.2	0.8
…	
0.4715	0.5285
0.1	0.9
0.1	0.9
…	
0.4715	0.5285
…	
0.99	0.01
0.1	0.9
…	
0.4715	0.5285

0.01	0.18
0.04	0.56
…	
0.4715^2	0.05285
0.01	0.72
0.01	0.45
…	
0.4715^2	0.5285*0.4
…	
0.99^2	0.007
0.01	0.27
…	
0.4715^2	0.05285

0.19
0.6
…
0.52437
0.73
0.46
…
0.68292
…
0.999
0.28
…
0.52437

0.19	0.6	…	0.52437
0.73	0.46	…	0.68292
…			
0.999	0.28	…	0.52437



Input:
batch_size by (1024+128=1152)

```
 0   1   2 … 1023 | 0  1  2 … 127
 1   1   2 … 1023 | 0  1  2 … 127
                …
99   1   2 … 1023 | 0  1  2 … 127
```

Each neuron has
A weight vector
[w_0 w_1 … w_{1151}]
and a bias:
b
and an activation f
(ReLU)

```
 0   1   2 … 4715  0   1   2 … 4715
 1   1   2 … 4715  0   1   2 … 4715
                  …
99   1   2 … 4715  0   1   2 … 4715
```

reshape

```
   0    0
   1    1
     …
4715  4715
     …
   1    1
   1    1
     …
4715 4715
     ..
  99  99
   1   1
     …
4715  44715
```

```
0.1     0.2
0.2     0.7
…
0.4715  0.1
0.1     0.8
0.1     0.5
…
0.4715  0.4
…
0.99    0.7
0.1     0.3
…
0.4715  0.1
```

```
0.1     0.9
0.2     0.8
…
0.4715  0.5285
0.1     0.9
0.1     0.9
…
0.4715  0.5285
…
0.99    0.01
0.1     0.9
…
0.4715  0.5285
```

*

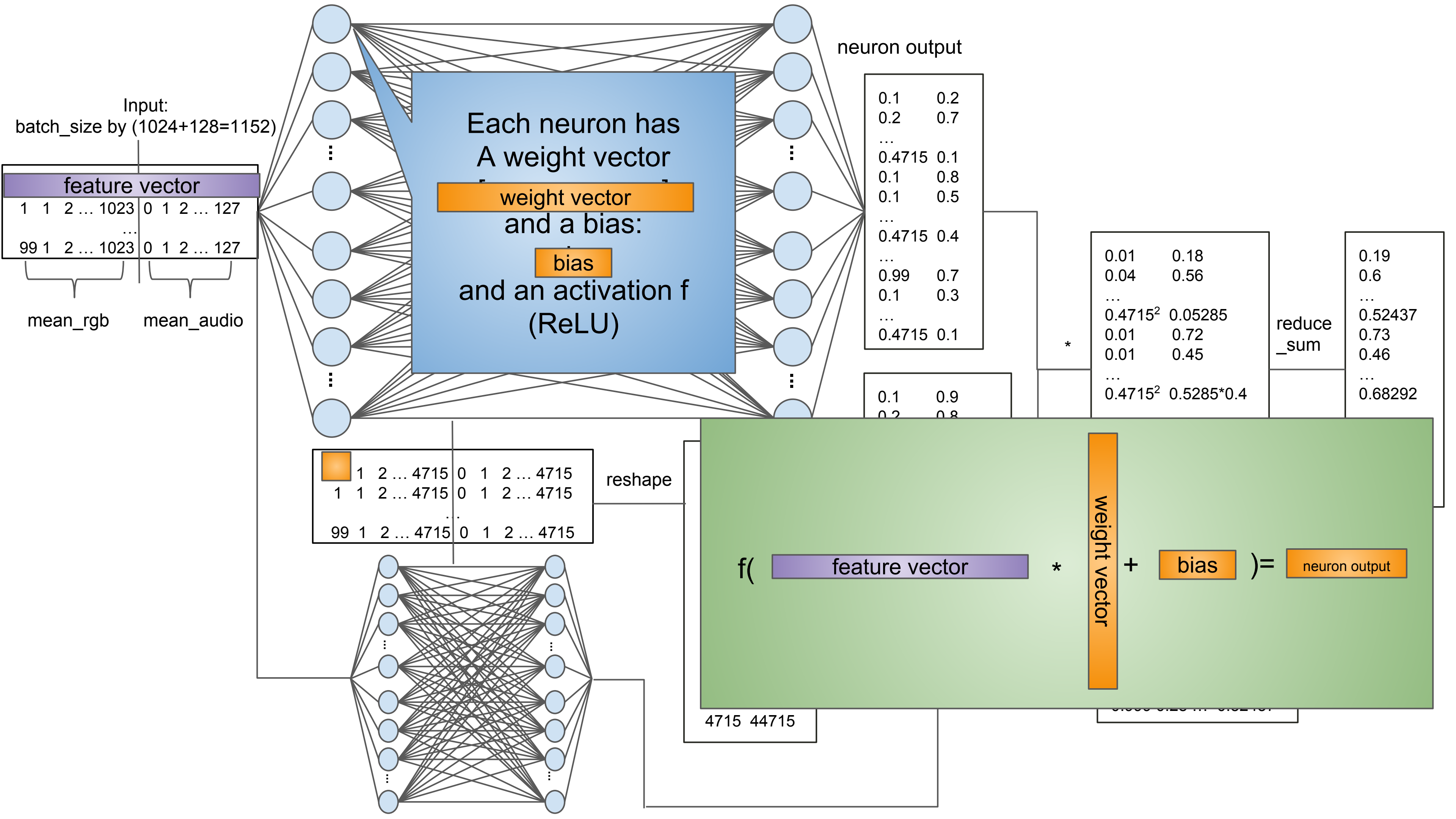
0.01 0.18
0.04 0.56
…
0.4715^2 0.05285
0.01 0.72
0.01 0.45
…
0.4715^2 0.5285*0.4
…
0.99^2 0.007
0.01 0.27
…
0.4715^2 0.05285

reduce
_sum

```
0.19
0.6
…
0.52437
0.73
0.46
…
0.68292
…
0.999
0.28
…
0.52437
```

reshape

```
0.19   0.6  … 0.52437
0.73   0.46 … 0.68292
…
0.999  0.28 … 0.52437
```



Each neuron has
A weight vector
weight vector
and a bias:
bias
and an activation f
(ReLU)

1 1 2 … 1023 | 0 1 2 … 127
…
99 1 2 … 1023 | 0 1 2 … 127

0.1 0.2
0.2 0.7
…
0.4715 0.1
0.1 0.8
0.1 0.5
…
0.4715 0.4
…
0.99 0.7
0.1 0.3
…
0.4715 0.1

0.1 0.9
0.2 0.8

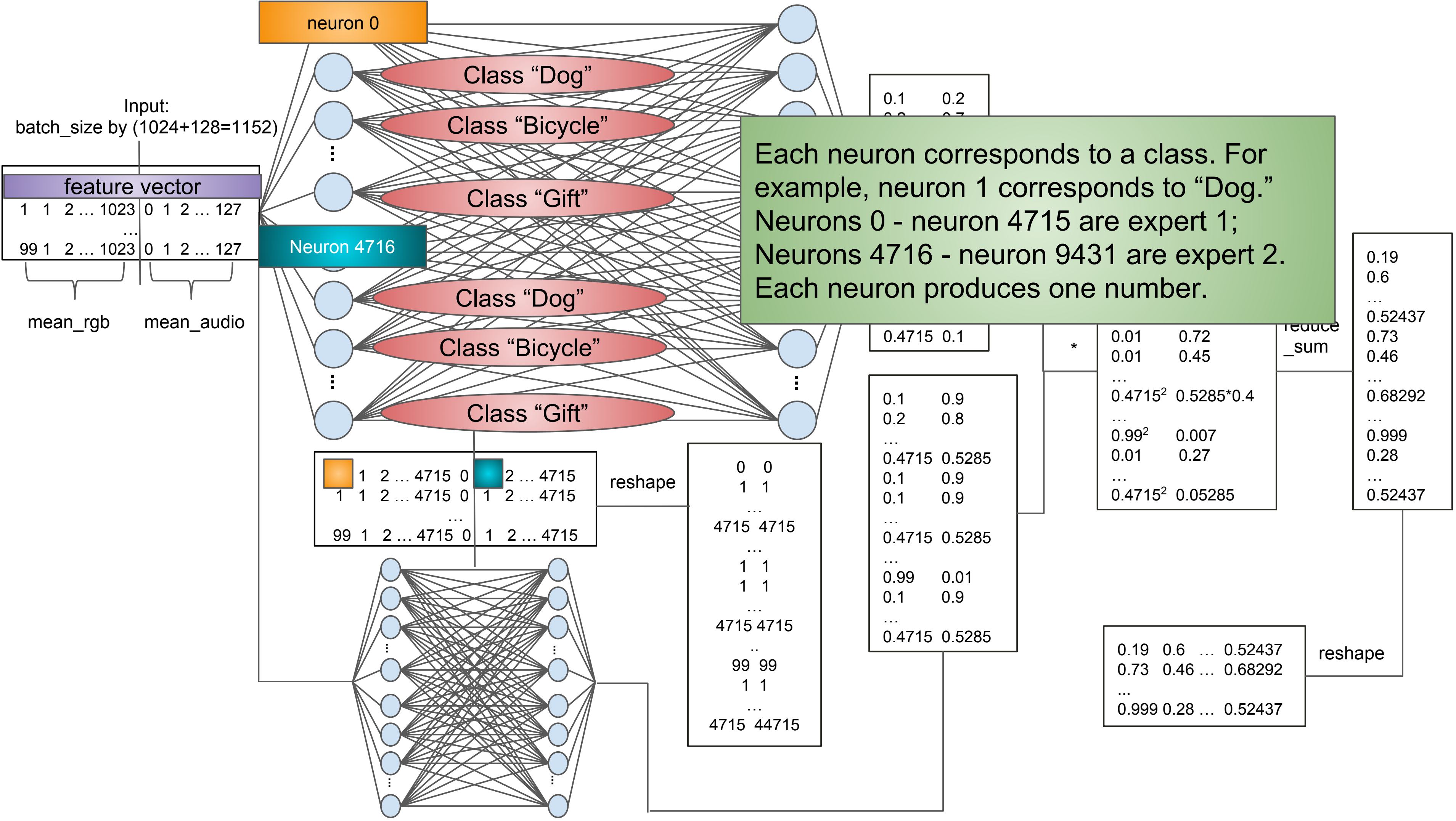
0.01 0.18
0.04 0.56
…
0.4715^2 0.05285
0.01 0.72
0.01 0.45
…
0.4715^2 0.5285*0.4

0.19
0.6
…
0.52437
0.73
0.46
…
0.68292

1 1 2 … 4715 | 0 1 2 … 4715
1 1 2 … 4715 | 0 1 2 … 4715
…
99 1 2 … 4715 | 0 1 2 … 4715

4715 44715

f(feature vector * weight vector + bias)= neuron output



1 1 2 … 1023 | 0 1 2 … 127
…
99 1 2 … 1023 | 0 1 2 … 127

1 2 … 4715 0 2 … 4715
1 1 2 … 4715 0 1 2 … 4715
…
99 1 2 … 4715 0 1 2 … 4715

0 0
1 1
…
4715 4715
…
1 1
1 1
…
4715 4715
..
99 99
1 1
…
4715 44715

0.1 0.2
0.4715 0.1

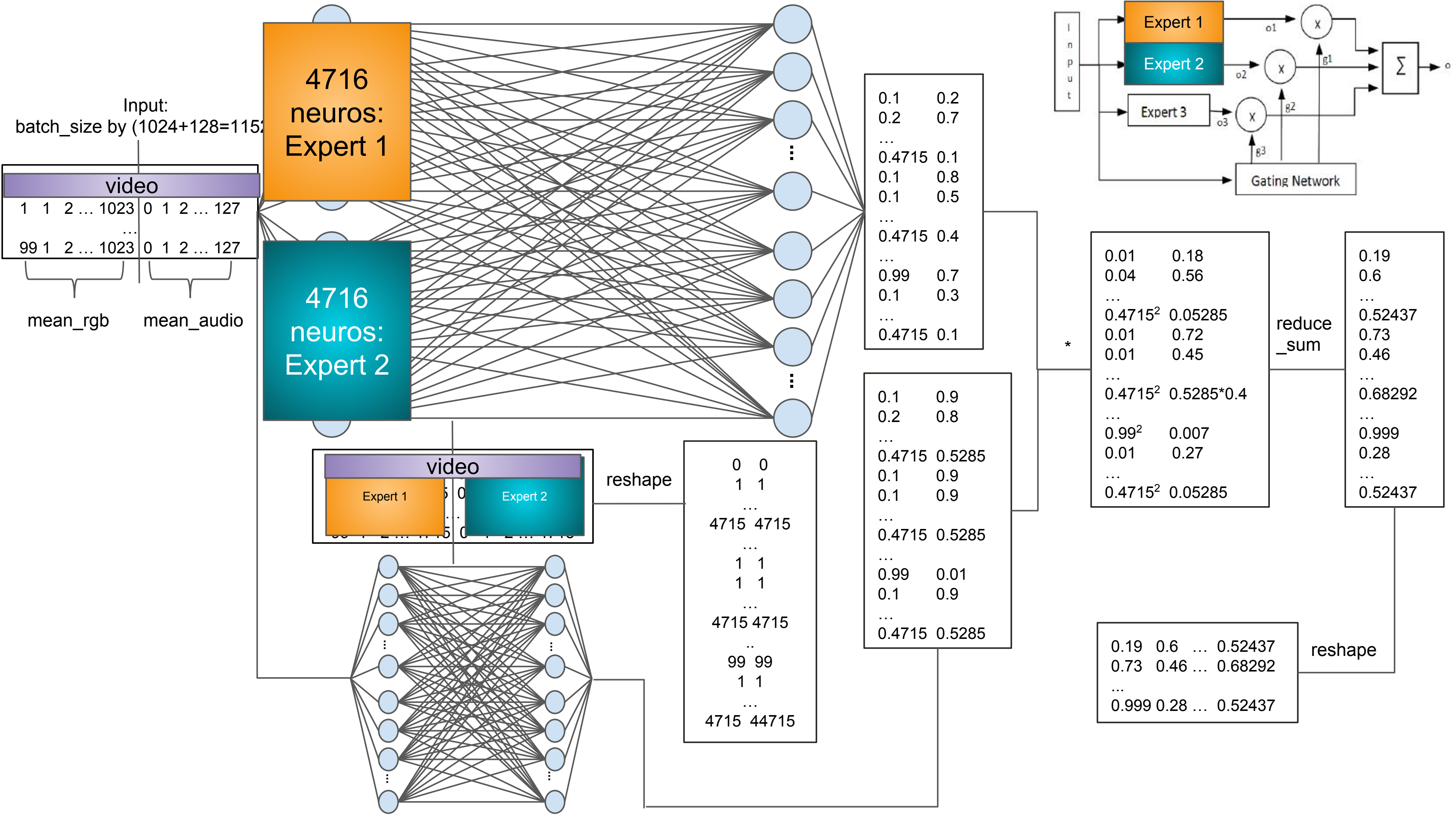
0.1 0.9
0.2 0.8
…
0.4715 0.5285
0.1 0.9
0.1 0.9
…
0.4715 0.5285
…
0.99 0.01
0.1 0.9
…
0.4715 0.5285

0.01 0.72
0.01 0.45
…
0.4715^2 0.5285*0.4
…
0.99^2 0.007
0.01 0.27
…
0.4715^2 0.05285

0.19
0.6
…
0.52437
0.73
0.46
…
0.68292
…
0.999
0.28
…
0.52437

0.19 0.6 … 0.52437
0.73 0.46 … 0.68292
…
0.999 0.28 … 0.52437

Each neuron corresponds to a class. For example, neuron 1 corresponds to “Dog.” Neurons 0 - neuron 4715 are expert 1; Neurons 4716 - neuron 9431 are expert 2. Each neuron produces one number.



Input:
batch_size by (1024+128=115

video

1 1 2 … 1023 | 0 1 2 … 127
…
99 1 2 … 1023 | 0 1 2 … 127

mean_rgb mean_audio

4716 neuros: Expert 1

4716 neuros: Expert 2

Expert 1

Expert 2

```
0.1     0.2
0.2     0.7
…
0.4715  0.1
0.1     0.8
0.1     0.5
…
0.4715  0.4
…
0.99    0.7
0.1     0.3
…
0.4715  0.1
```

video

Expert 1 Expert 2

reshape

```
  0     0
  1     1
  …
4715  4715
  …
  1     1
  1     1
  …
4715 4715
  ..
 99   99
  1    1
  …
4715  44715
```

```
0.1     0.9
0.2     0.8
…
0.4715  0.5285
0.1     0.9
0.1     0.9
…
0.4715  0.5285
…
0.99    0.01
0.1     0.9
…
0.4715  0.5285
```

*

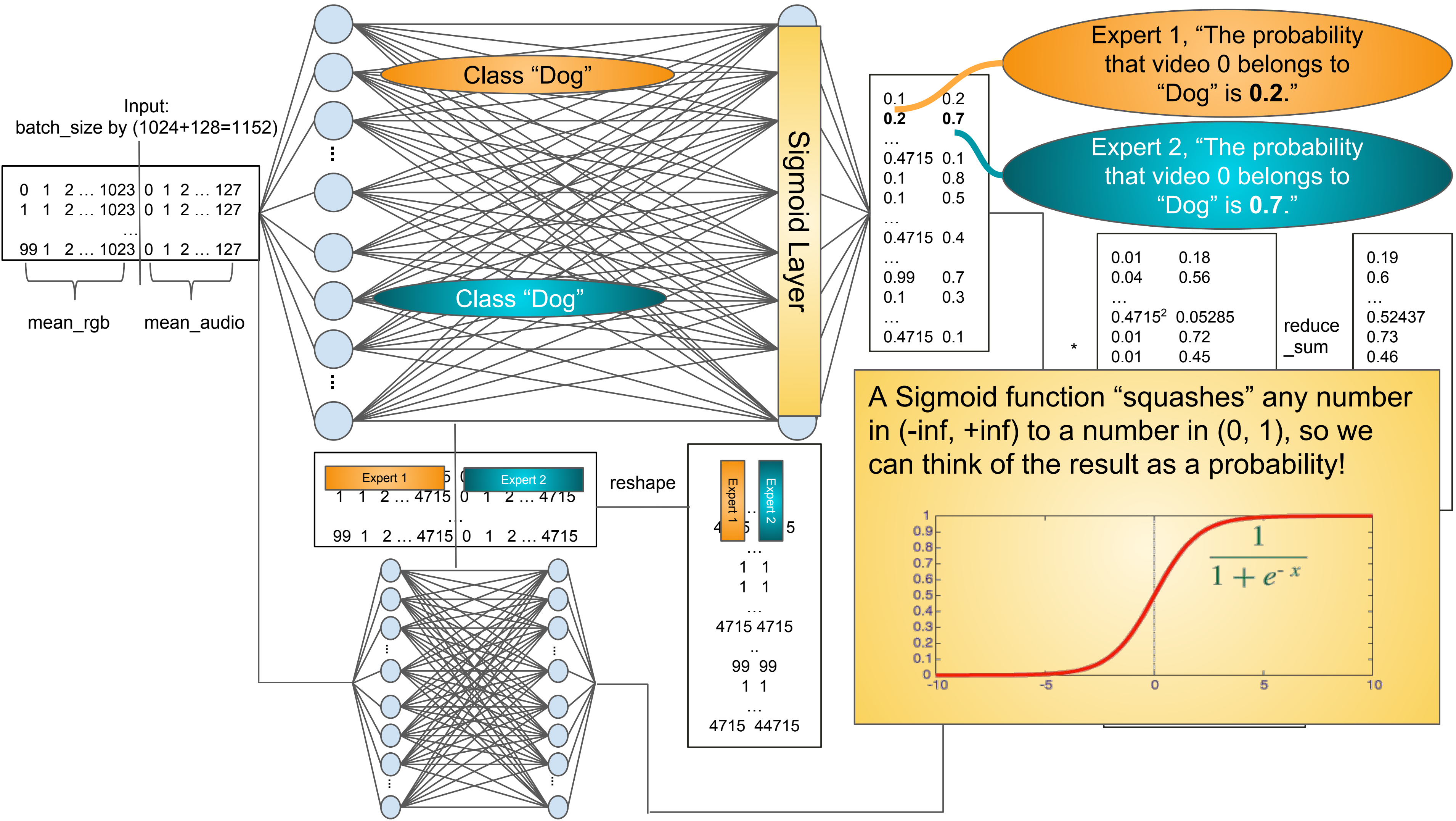
```
0.01       0.18
0.04       0.56
…
0.4715^2   0.05285
0.01       0.72
0.01       0.45
…
0.4715^2   0.5285*0.4
…
0.99^2     0.007
0.01       0.27
…
0.4715^2   0.05285
```

reduce _sum

```
0.19
0.6
…
0.52437
0.73
0.46
…
0.68292
…
0.999
0.28
…
0.52437
```

reshape

```
0.19   0.6   …  0.52437
0.73   0.46  …  0.68292
…
0.999  0.28  …  0.52437
```



0 1 2 … 1023 | 0 1 2 … 127
1 1 2 … 1023 | 0 1 2 … 127
…
99 1 2 … 1023 | 0 1 2 … 127

Expert 1, "The probability that video 0 belongs to "Dog" is **0.2**."

Expert 2, "The probability that video 0 belongs to "Dog" is **0.7**."

0.1 0.2
0.2 0.7
…
0.4715 0.1
0.1 0.8
0.1 0.5
…
0.4715 0.4
…
0.99 0.7
0.1 0.3
…
0.4715 0.1

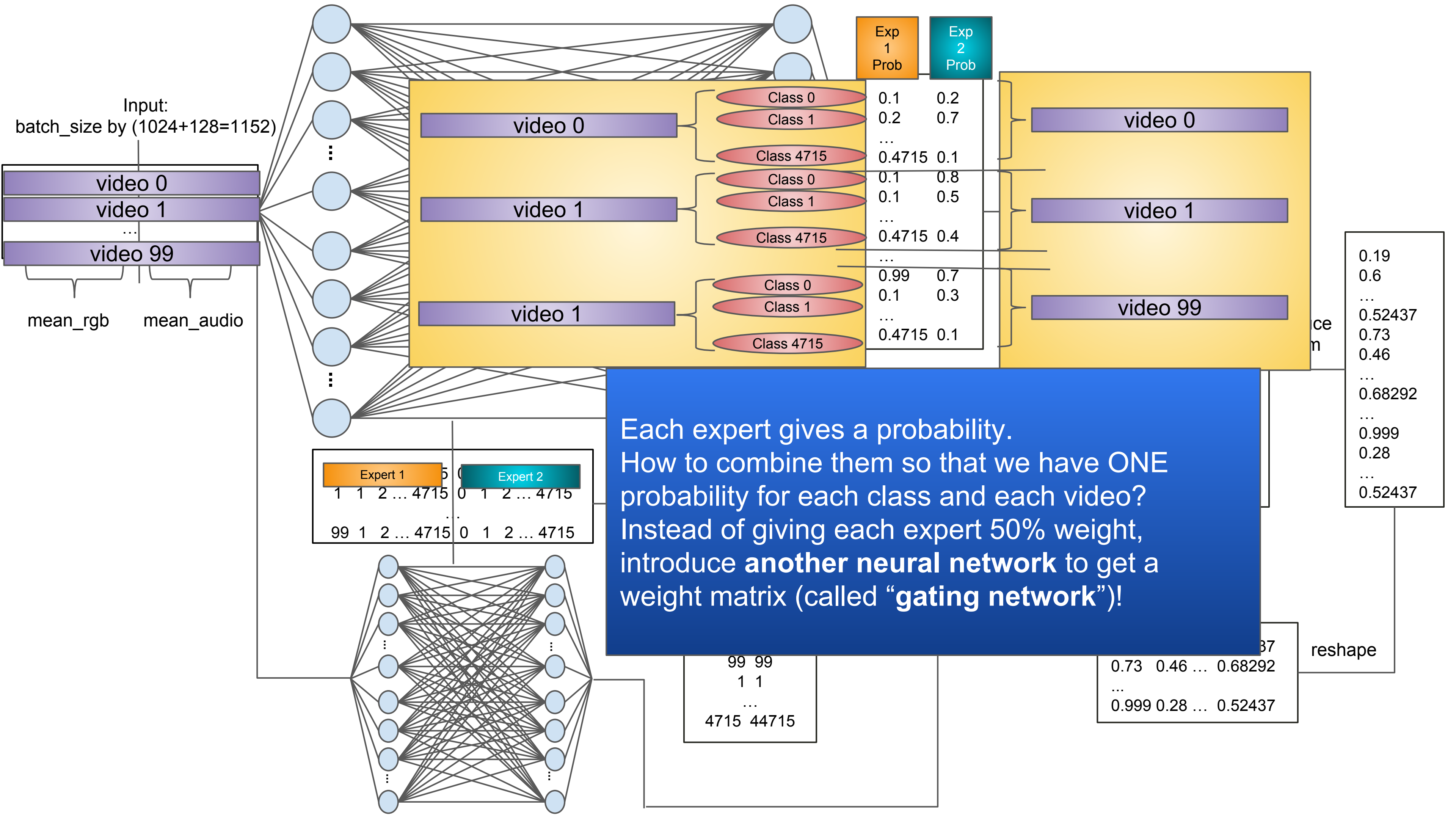
0.01 0.18
0.04 0.56
…
0.4715^2 0.05285
0.01 0.72
0.01 0.45

0.19
0.6
…
0.52437
0.73
0.46

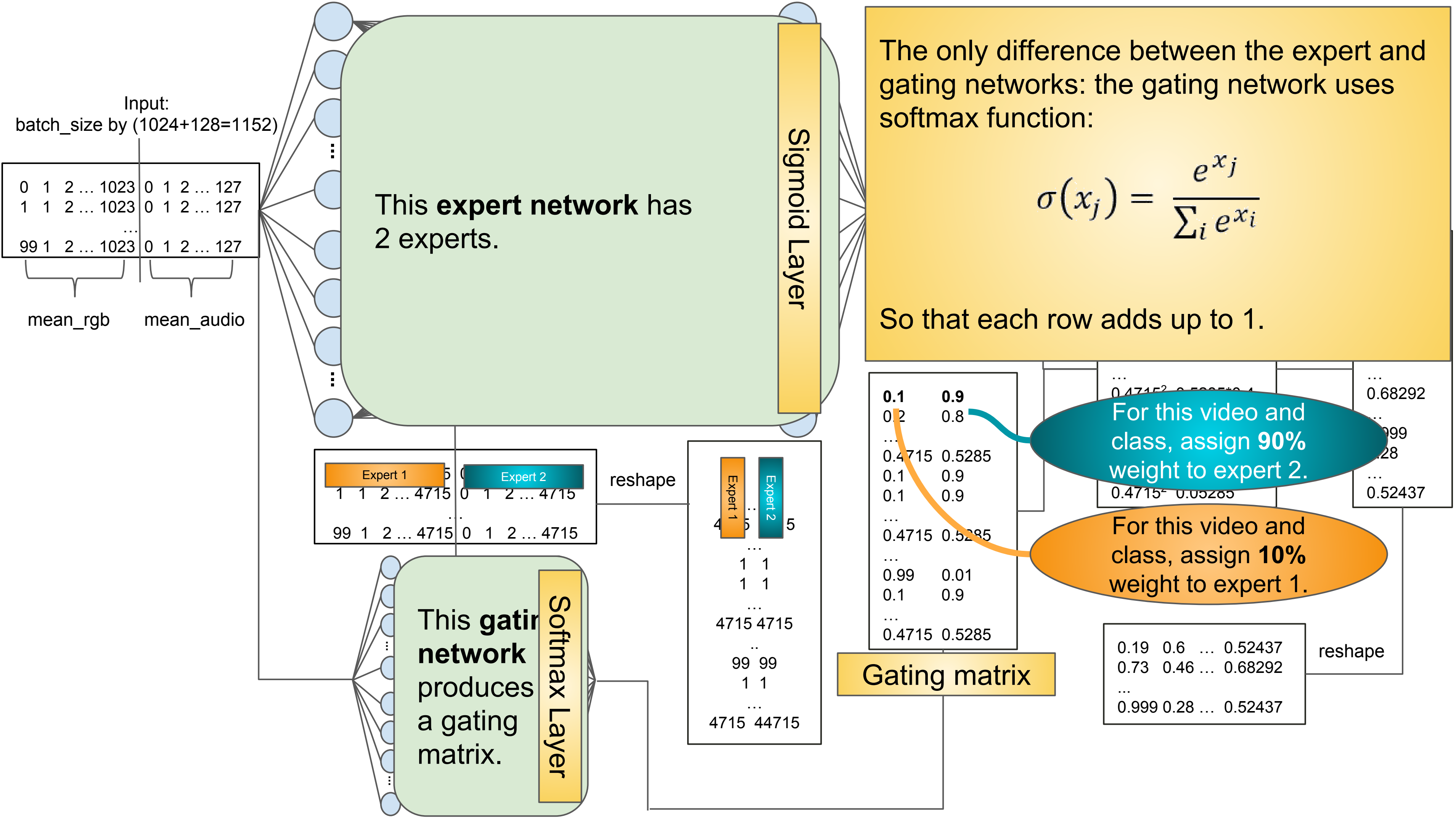
5 (
1 1 2 … 4715 0 1 2 … 4715
…
99 1 2 … 4715 0 1 2 … 4715

4 .. 5 5
…
1 1
1 1
…
4715 4715
..
99 99
1 1
…
4715 44715

A Sigmoid function "squashes" any number in (-inf, +inf) to a number in (0, 1), so we can think of the result as a probability!



Each expert gives a probability.
How to combine them so that we have ONE probability for each class and each video?
Instead of giving each expert 50% weight, introduce **another neural network** to get a weight matrix (called “**gating network**”)!



This **expert network** has 2 experts.

The only difference between the expert and gating networks: the gating network uses softmax function:

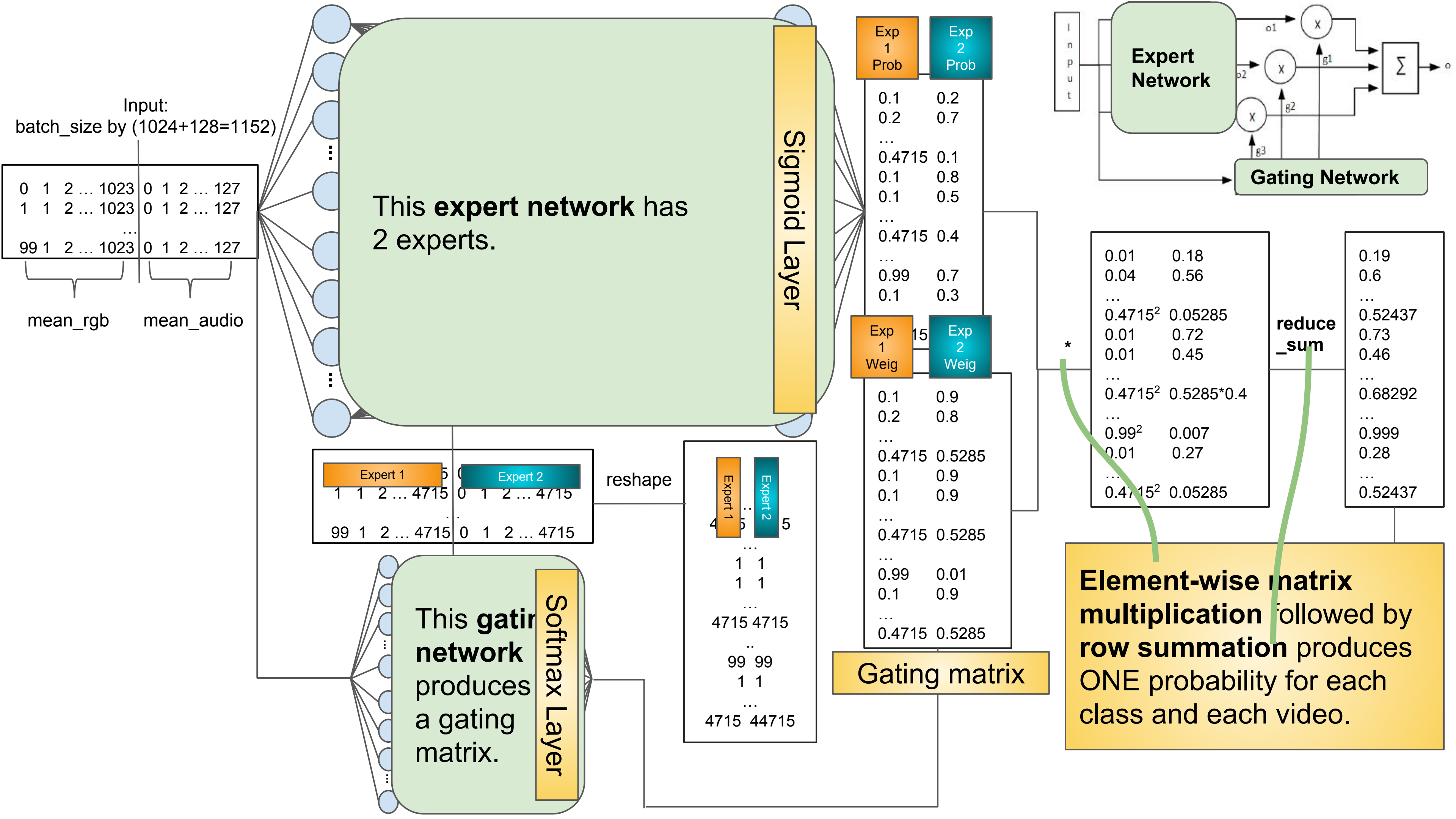
$$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}}$$

So that each row adds up to 1.

For this video and class, assign **90%** weight to expert 2.

For this video and class, assign **10%** weight to expert 1.

This **gatir network** produces a gating matrix.



Input:
batch_size by (1024+128=1152)

```
0   1   2 … 1023  0  1  2 … 127
1   1   2 … 1023  0  1  2 … 127
                …
99  1   2 … 1023  0  1  2 … 127
```

mean_rgb mean_audio

This **expert network** has 2 experts.

Sigmoid Layer

Exp 1 Prob	Exp 2 Prob
0.1	0.2
0.2	0.7
…	
0.4715	0.1
0.1	0.8
0.1	0.5
…	
0.4715	0.4
…	
0.99	0.7
0.1	0.3

Exp 1 Weig	Exp 2 Weig
0.1	0.9
0.2	0.8
…	
0.4715	0.5285
0.1	0.9
0.1	0.9
…	
0.4715	0.5285
…	
0.99	0.01
0.1	0.9
…	
0.4715	0.5285

Gating matrix

```
Expert 1              Expert 2
1   1   2 … 4715  0   1   2 … 4715
               …
99  1   2 … 4715  0   1   2 … 4715
```

reshape

```
Expert 1  Expert 2
4 .. 5         5
     …
   1   1
   1   1
     …
4715 4715
     ..
  99  99
   1   1
     …
4715  44715
```

This **gating network** produces a gating matrix.

Softmax Layer

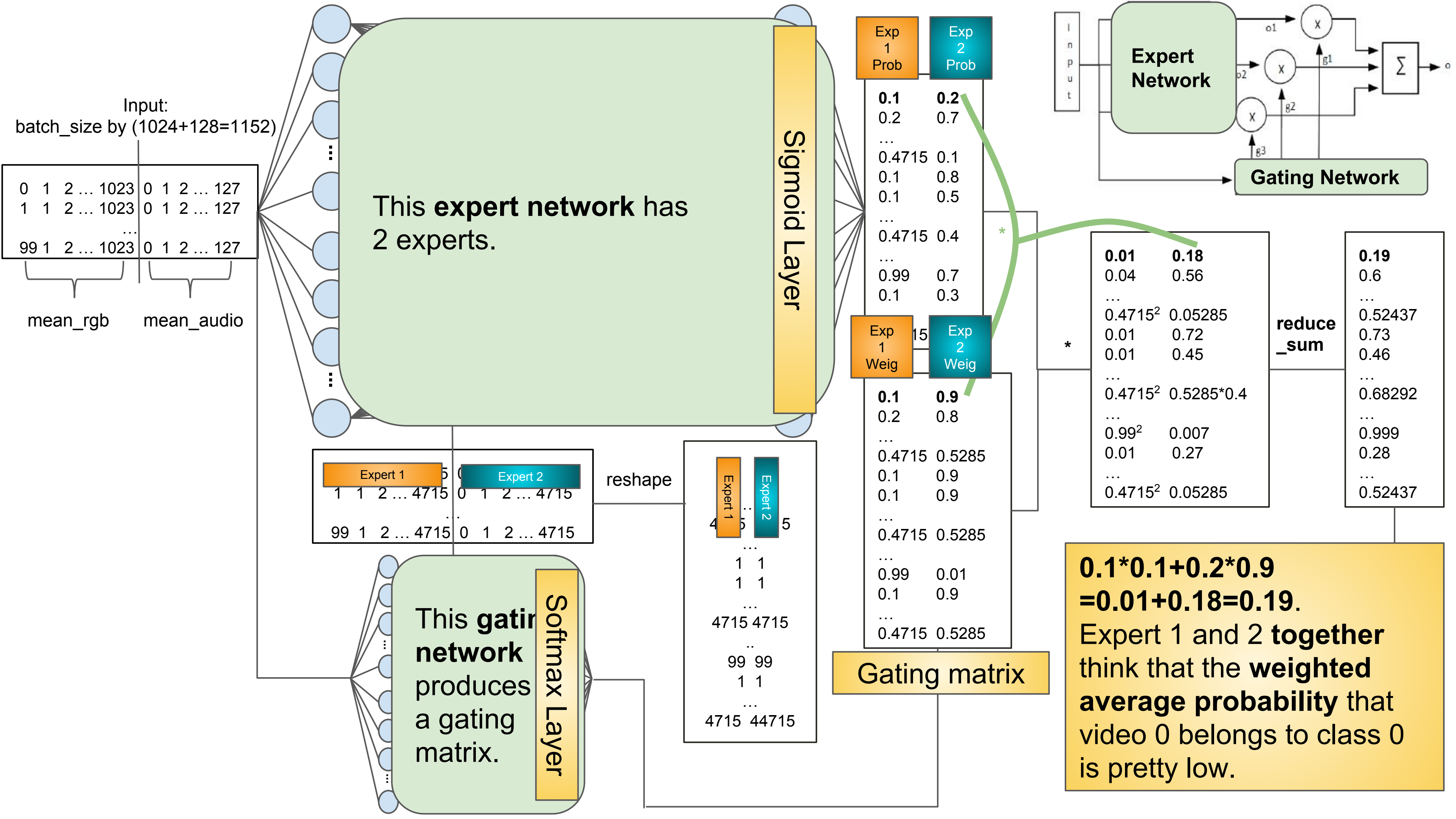
*

0.01	0.18
0.04	0.56
…	
0.4715^2	0.05285
0.01	0.72
0.01	0.45
…	
0.4715^2	0.5285*0.4
…	
0.99^2	0.007
0.01	0.27
…	
0.4715^2	0.05285

reduce_sum

```
0.19
0.6
…
0.52437
0.73
0.46
…
0.68292
…
0.999
0.28
…
0.52437
```

Element-wise matrix multiplication followed by **row summation** produces ONE probability for each class and each video.



```
0   1   2 … 1023 | 0  1  2 … 127
1   1   2 … 1023 | 0  1  2 … 127
                …
99  1   2 … 1023 | 0  1  2 … 127
```

This **expert network** has 2 experts.

This **gatir network** produces a gating matrix.

```
 1   1   2 … 4715  0   1   2 … 4715
                 …
99   1   2 … 4715  0   1   2 … 4715
```

```
4 5   5
  …
 1   1
 1   1
  …
4715 4715
  ..
 99  99
 1   1
  …
4715  44715
```

```
0.1     0.2
0.2     0.7
…
0.4715  0.1
0.1     0.8
0.1     0.5
…
0.4715  0.4
…
0.99    0.7
0.1     0.3
```

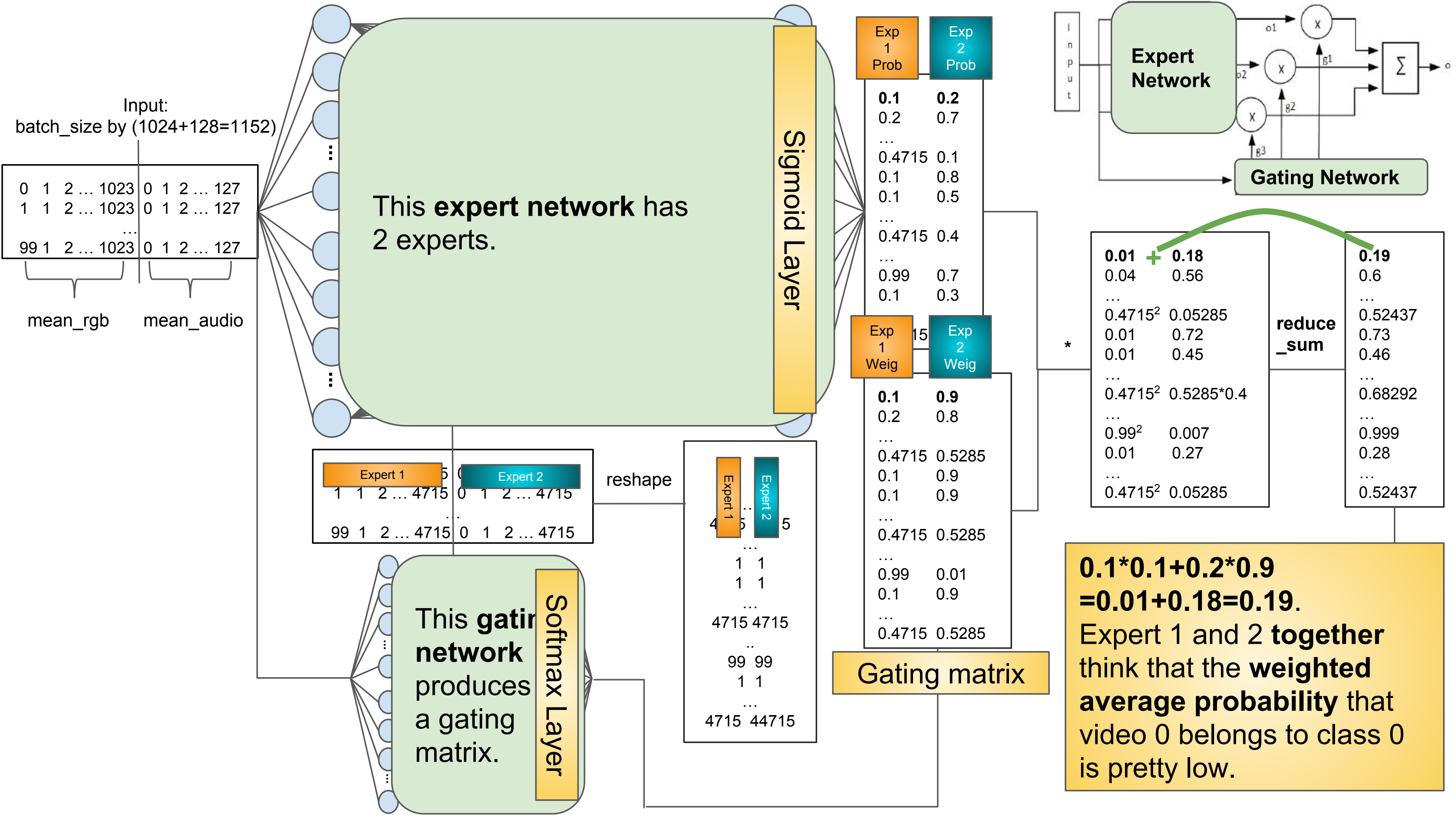
```
0.1     0.9
0.2     0.8
…
0.4715  0.5285
0.1     0.9
0.1     0.9
…
0.4715  0.5285
…
0.99    0.01
0.1     0.9
…
0.4715  0.5285
```

Gating matrix

```
0.01         0.18
0.04         0.56
…
0.4715^2     0.05285
0.01         0.72
0.01         0.45
…
0.4715^2     0.5285*0.4
…
0.99^2       0.007
0.01         0.27
…
0.4715^2     0.05285
```

```
0.19
0.6
…
0.52437
0.73
0.46
…
0.68292
…
0.999
0.28
…
0.52437
```

0.1*0.1+0.2*0.9 =0.01+0.18=0.19. Expert 1 and 2 **together** think that the **weighted average probability** that video 0 belongs to class 0 is pretty low.



Input:
batch_size by (1024+128=1152)

0 1 2 … 1023	0 1 2 … 127
1 1 2 … 1023	0 1 2 … 127
…	
99 1 2 … 1023	0 1 2 … 127
mean_rgb	mean_audio

This **expert network** has 2 experts.

Sigmoid Layer

Expert 1 5	(Expert 2
1 1 2 … 4715	0 1 2 … 4715
…	
99 1 2 … 4715	0 1 2 … 4715

reshape

Expert 1	Expert 2
4 .. 5	5
…	
1	1
1	1
…	
4715	4715
..	
99	99
1	1
…	
4715	44715

This gatir **network** produces a gating matrix.

Softmax Layer

Exp 1 Prob	Exp 2 Prob
0.1	**0.2**
0.2	0.7
…	
0.4715	0.1
0.1	0.8
0.1	0.5
…	
0.4715	0.4
…	
0.99	0.7
0.1	0.3

Exp 1 Weig 15	Exp 2 Weig
0.1	**0.9**
0.2	0.8
…	
0.4715	0.5285
0.1	0.9
0.1	0.9
…	
0.4715	0.5285
…	
0.99	0.01
0.1	0.9
…	
0.4715	0.5285

Gating matrix

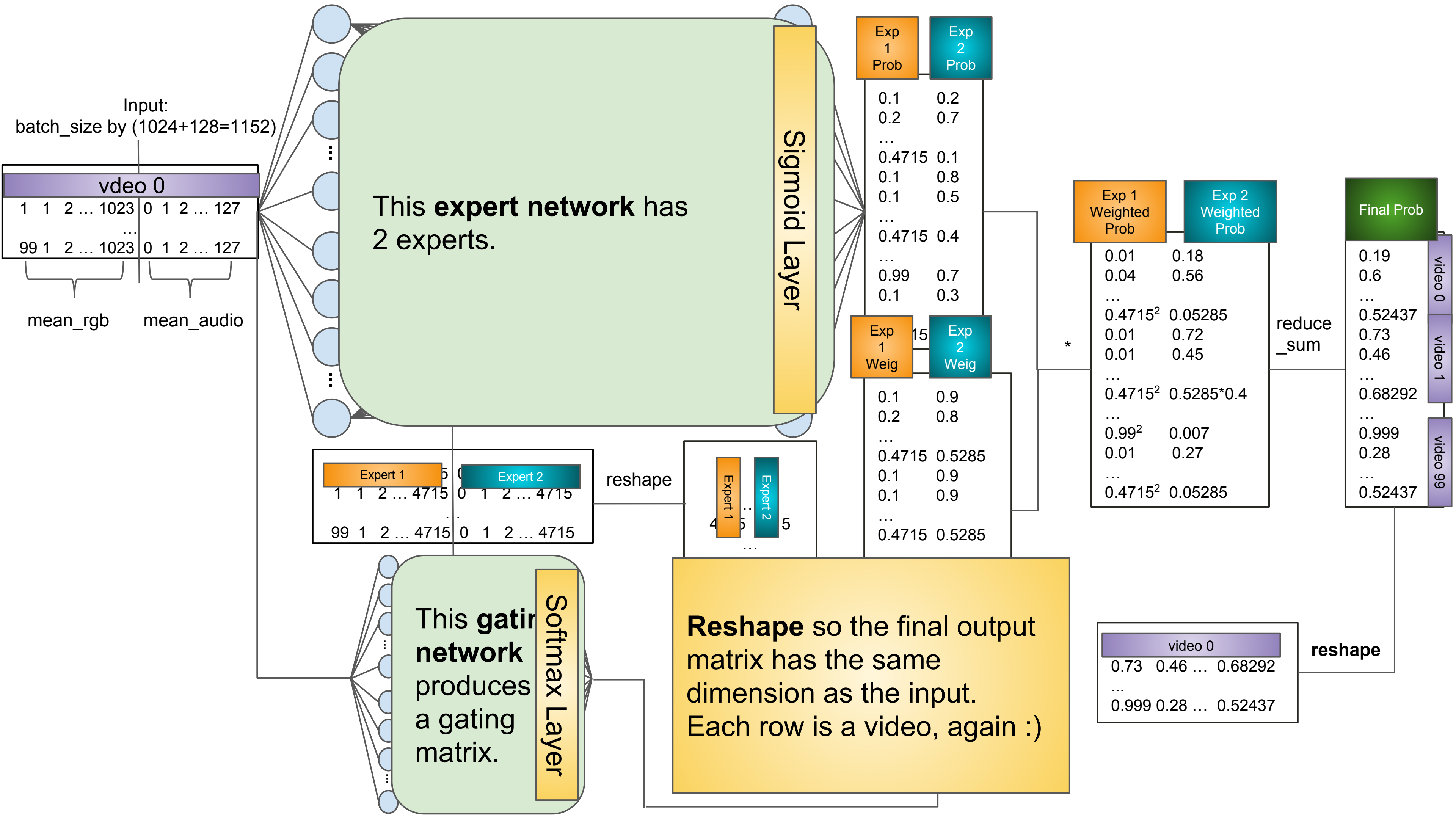
*

0.01 +	**0.18**
0.04	0.56
…	
0.4715^2	0.05285
0.01	0.72
0.01	0.45
…	
0.4715^2	0.5285*0.4
…	
0.99^2	0.007
0.01	0.27
…	
0.4715^2	0.05285

reduce_sum

0.19
0.6
…
0.52437
0.73
0.46
…
0.68292
…
0.999
0.28
…
0.52437

0.1*0.1+0.2*0.9 =0.01+0.18=0.19. Expert 1 and 2 **together** think that the **weighted average probability** that video 0 belongs to class 0 is pretty low.



Input:
batch_size by (1024+128=1152)

vdeo 0

1 1 2 … 1023 | 0 1 2 … 127
…
99 1 2 … 1023 | 0 1 2 … 127

mean_rgb mean_audio

This **expert network** has 2 experts.

Sigmoid Layer

Exp 1 Prob	Exp 2 Prob
0.1	0.2
0.2	0.7
…	
0.4715	0.1
0.1	0.8
0.1	0.5
…	
0.4715	0.4
…	
0.99	0.7
0.1	0.3

Exp 1 Weig	Exp 2 Weig
0.1	0.9
0.2	0.8
…	
0.4715	0.5285
0.1	0.9
0.1	0.9
…	
0.4715	0.5285

*

Exp 1 Weighted Prob	Exp 2 Weighted Prob
0.01	0.18
0.04	0.56
…	
0.4715^2	0.05285
0.01	0.72
0.01	0.45
…	
0.4715^2	0.5285*0.4
…	
0.99^2	0.007
0.01	0.27
…	
0.4715^2	0.05285

reduce_sum

Final Prob	
0.19	video 0
0.6	
…	
0.52437	
0.73	video 1
0.46	
…	
0.68292	
…	
0.999	video 99
0.28	
…	
0.52437	

reshape

video 0
0.73 0.46 … 0.68292
…
0.999 0.28 … 0.52437

Expert 1 Expert 2
1 1 2 … 4715 0 1 2 … 4715
…
99 1 2 … 4715 0 1 2 … 4715

reshape

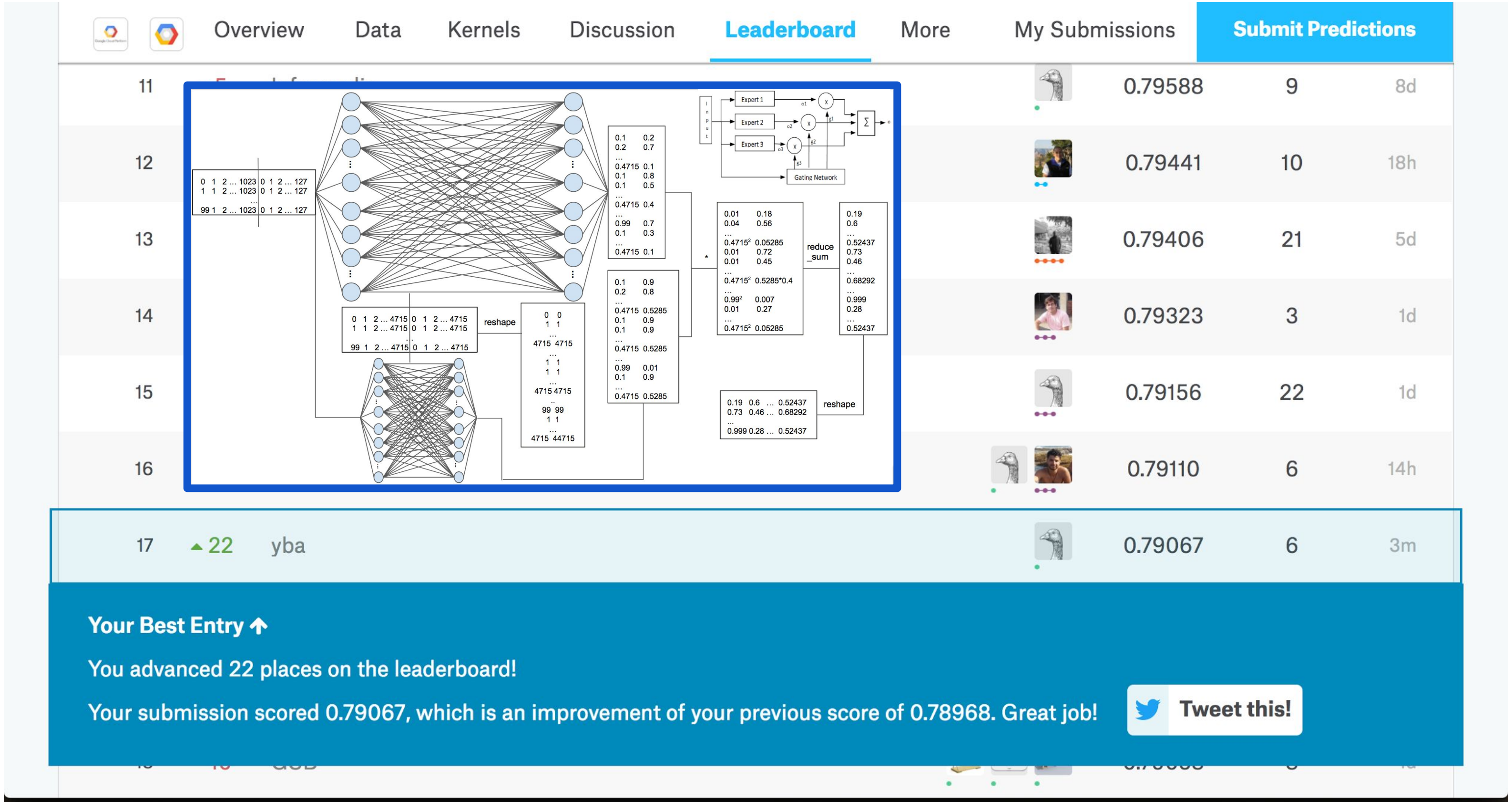
Expert 1 Expert 2
4 … 5 … 5
…

This **gatir** **network** produces a gating matrix.

Softmax Layer

Reshape so the final output matrix has the same dimension as the input.
Each row is a video, again :)

Mixture of Expert (MOE) model gives our current best result.



```
0  1  2 … 1023 | 0  1  2 … 127
1  1  2 … 1023 | 0  1  2 … 127
…
99 1  2 … 1023 | 0  1  2 … 127
```

```
0  1  2 … 4715 | 0  1  2 … 4715
1  1  2 … 4715 | 0  1  2 … 4715
…
99 1  2 … 4715 | 0  1  2 … 4715
```

```
0  0
1  1
…
4715 4715
…
1  1
1  1
…
4715 4715
..
99  99
1  1
…
4715  44715
```

```
0.1     0.2
0.2     0.7
…
0.4715  0.1
0.1     0.8
0.1     0.5
…
0.4715  0.4
…
0.99    0.7
0.1     0.3
…
0.4715  0.1
```

```
0.1     0.9
0.2     0.8
…
0.4715  0.5285
0.1     0.9
0.1     0.9
…
0.4715  0.5285
…
0.99    0.01
0.1     0.9
…
0.4715  0.5285
```

```
0.01     0.18
0.04     0.56
…
0.4715²  0.05285
0.01     0.72
0.01     0.45
…
0.4715²  0.5285*0.4
…
0.99²    0.007
0.01     0.27
…
0.4715²  0.05285
```

```
0.19
0.6
…
0.52437
0.73
0.46
…
0.68292
…
0.999
0.28
…
0.52437
```

```
0.19  0.6  … 0.52437
0.73  0.46 … 0.68292
…
0.999 0.28 … 0.52437
```

11			0.79588	9	8d
12			0.79441	10	18h
13			0.79406	21	5d
14			0.79323	3	1d
15			0.79156	22	1d
16			0.79110	6	14h
17	▲ 22	yba	0.79067	6	3m

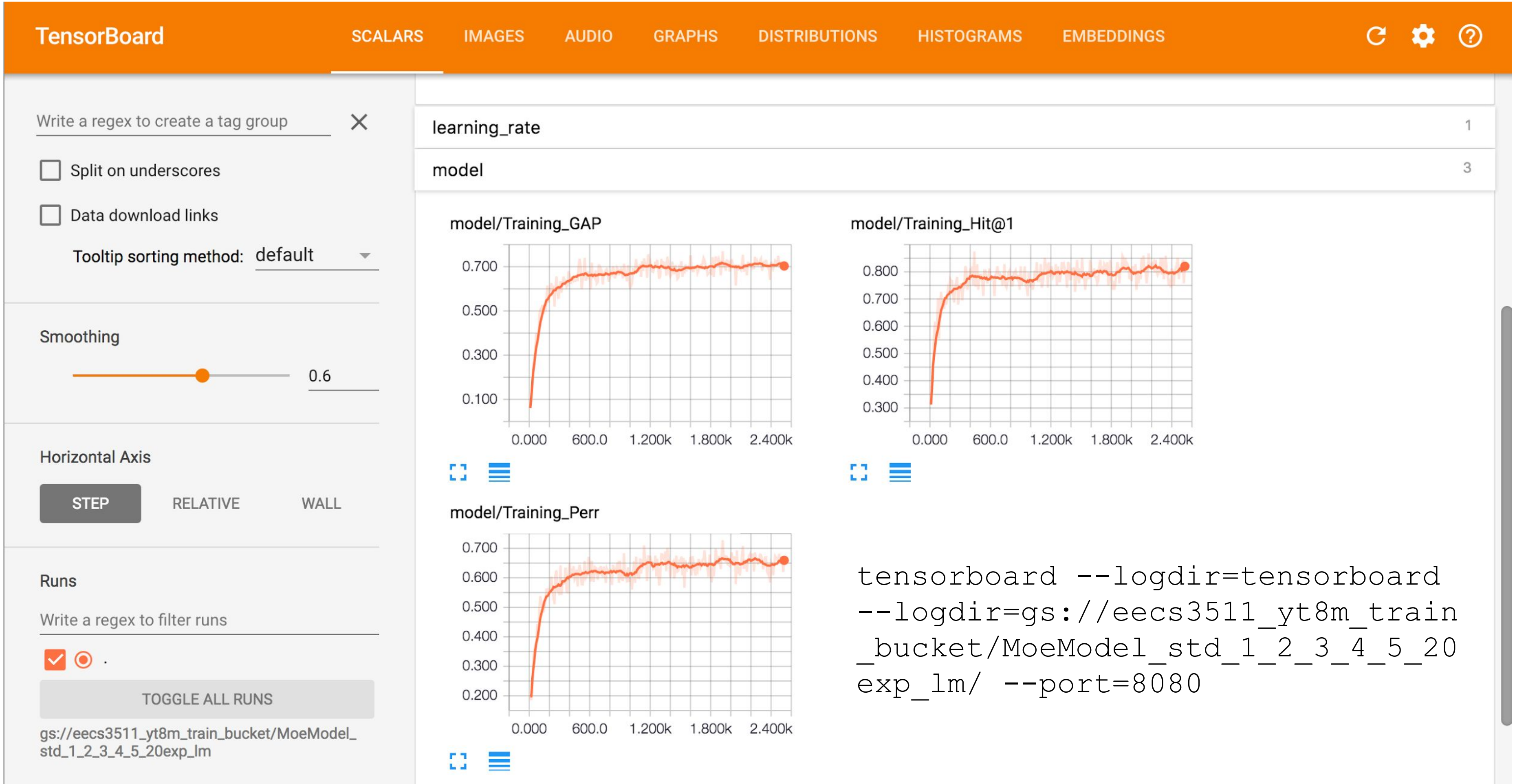
Your Best Entry ↑

You advanced 22 places on the leaderboard!

Your submission scored 0.79067, which is an improvement of your previous score of 0.78968. Great job!

Tweet this!

Results



Write a regex to create a tag group ✕

□ Split on underscores

□ Data download links

Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis: [✓ STEP] RELATIVE WALL

Runs

Write a regex to filter runs

☑ ◉ .

gs://eecs3511_yt8m_train_bucket/MoeModel_std_1_2_3_4_5_20exp_lm

learning_rate	1
model	3

```
tensorboard --logdir=tensorboard
--logdir=gs://eecs3511_yt8m_train
_bucket/MoeModel_std_1_2_3_4_5_20
exp_lm/ --port=8080
```

Links

- Dataset: https://research.google.com/youtube8m/
- Kaggle: https://www.kaggle.com/c/youtube8m
- Github: https://github.com/yunshengb/youtube-8m
- Paper: https://arxiv.org/pdf/1609.08675.pdf
- Paper Reading: https://docs.google.com/document/d/1zYSWcDrX38v2glNRXyKmdLDINnkPCHrp01gyqaet2fs/edit?usp=sharing