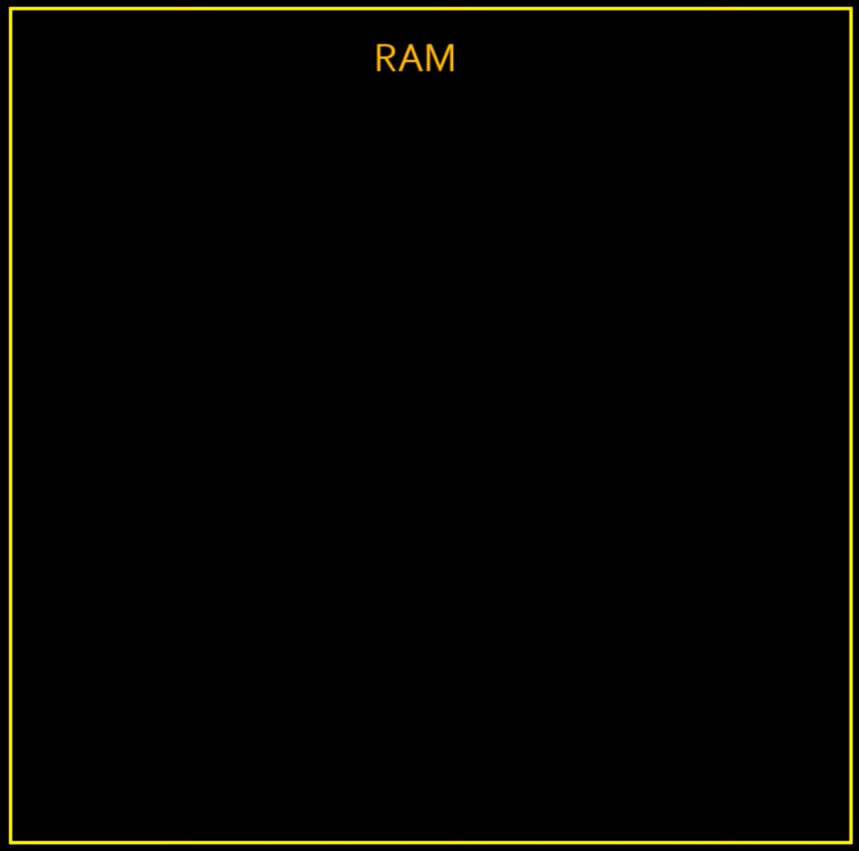
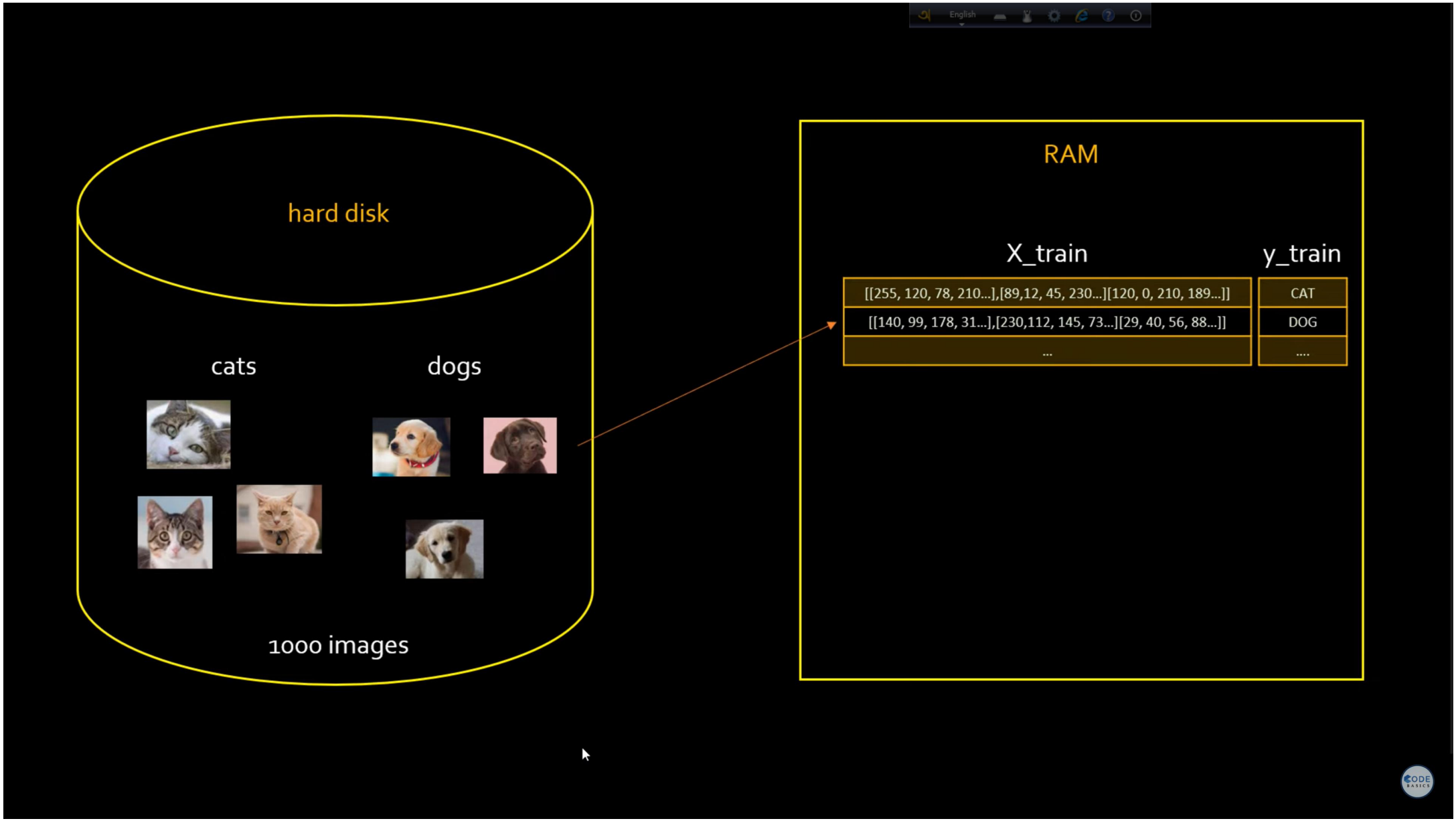
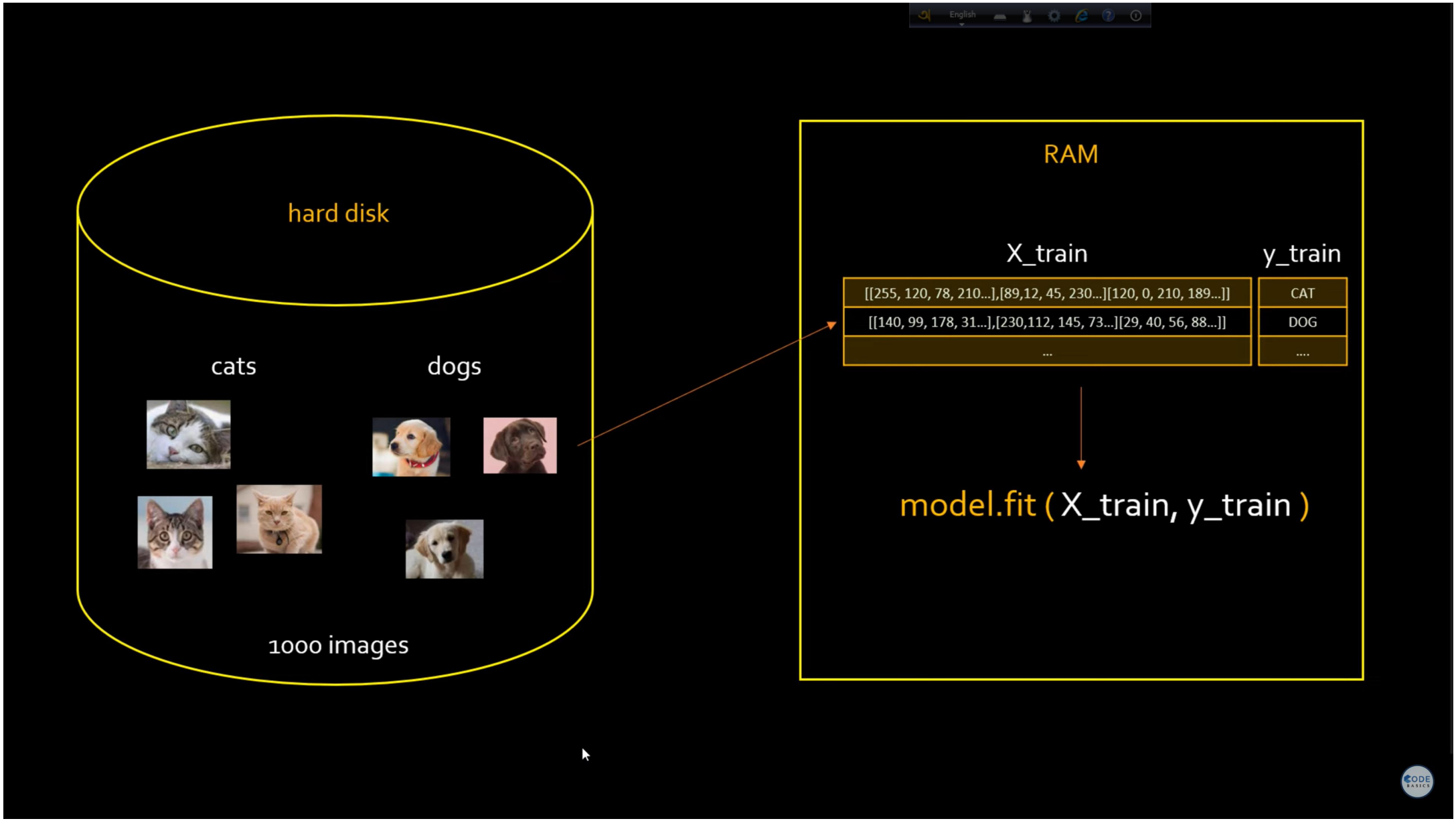
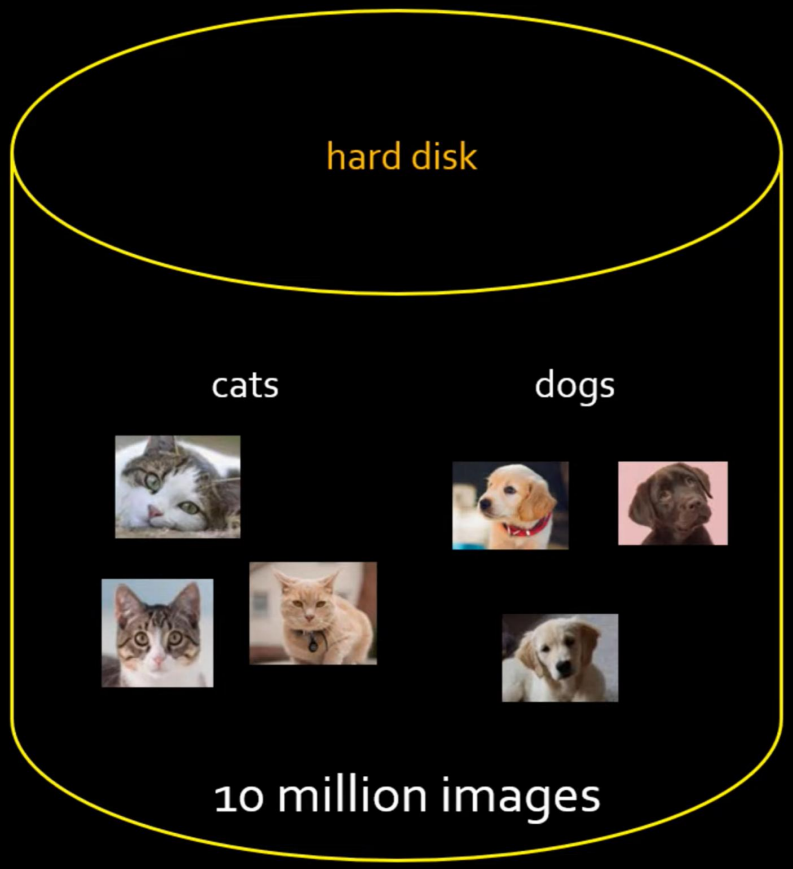

hard disk
cats
dogs
1000 images

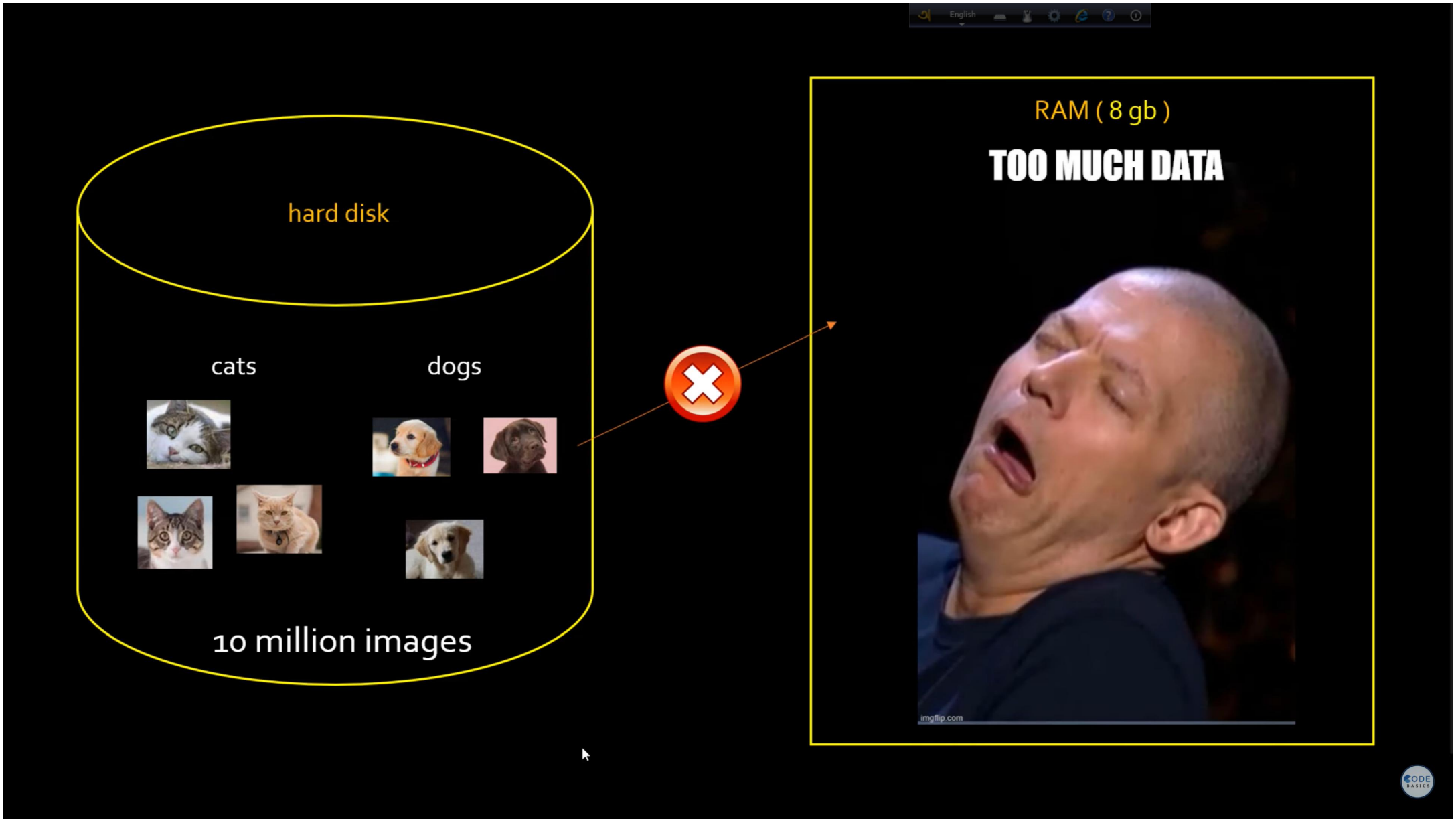

RAM

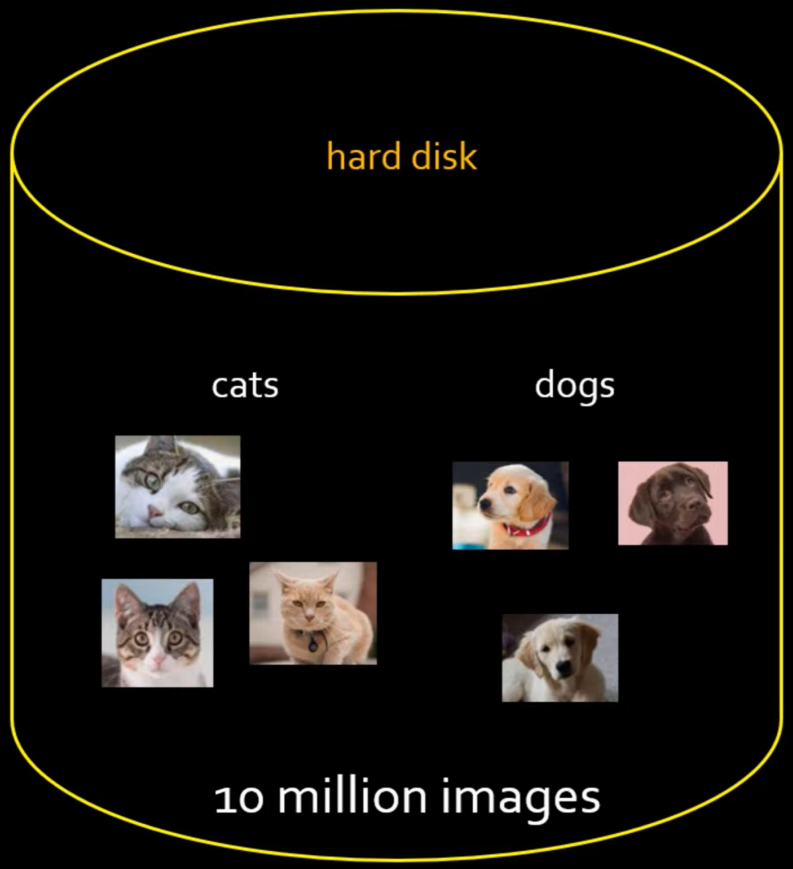

hard disk
cats
dogs
1000 images
RAM
X_train
y_train
[[255, 120, 78, 210...],[89,12, 45, 230...][120, 0, 210, 189...]]
CAT
[[140, 99, 178, 31...],[230,112, 145, 73...][29, 40, 56, 88...]]
DOG
...
....


hard disk
cats
dogs
1000 images
RAM
X_train
y_train
[[255, 120, 78, 210...],[89,12, 45, 230...][120, 0, 210, 189...]]
CAT
[[140, 99, 178, 31...],[230,112, 145, 73...][29, 40, 56, 88...]]
DOG
...
....
model.fit (X_train, y_train)

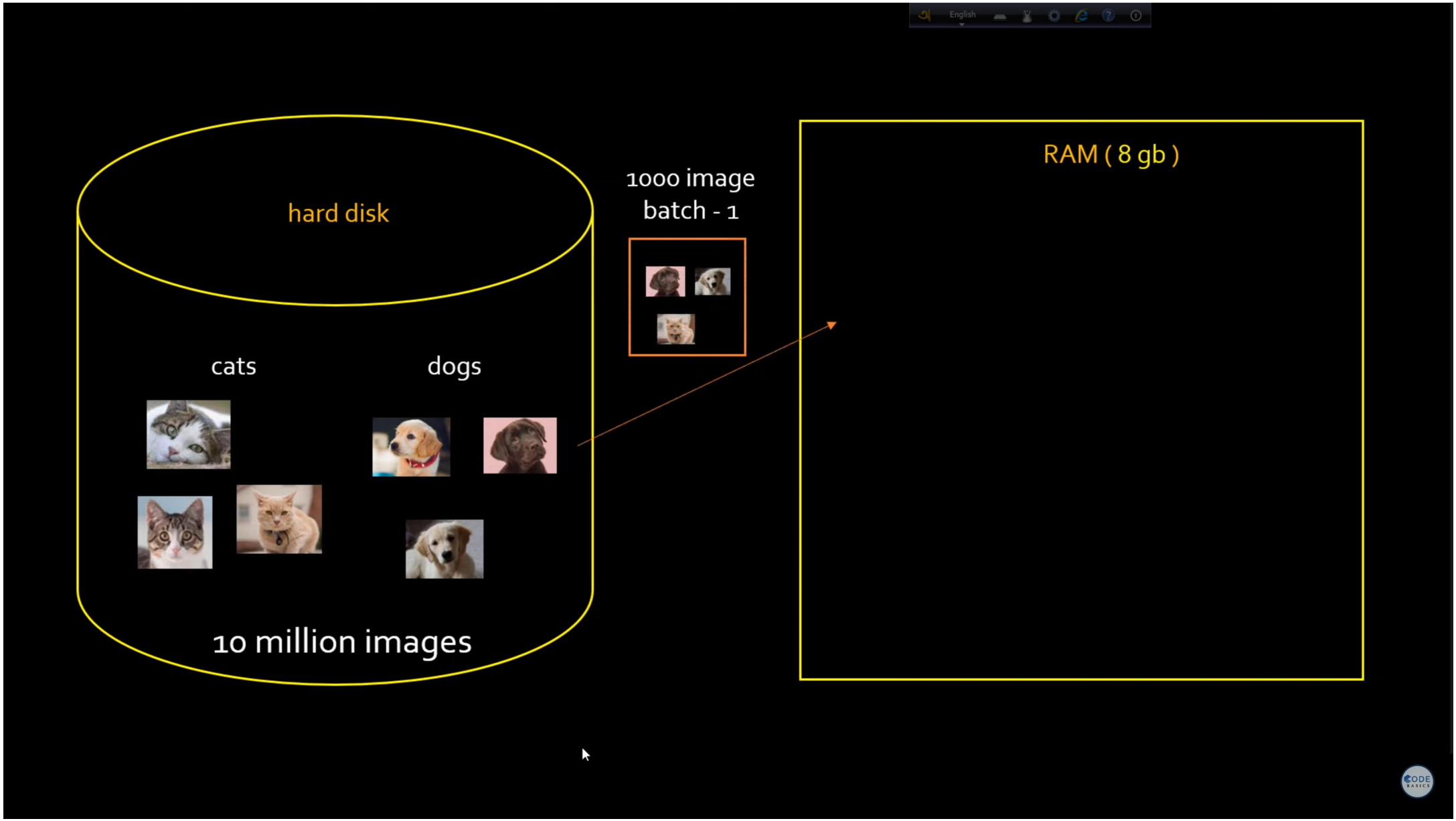


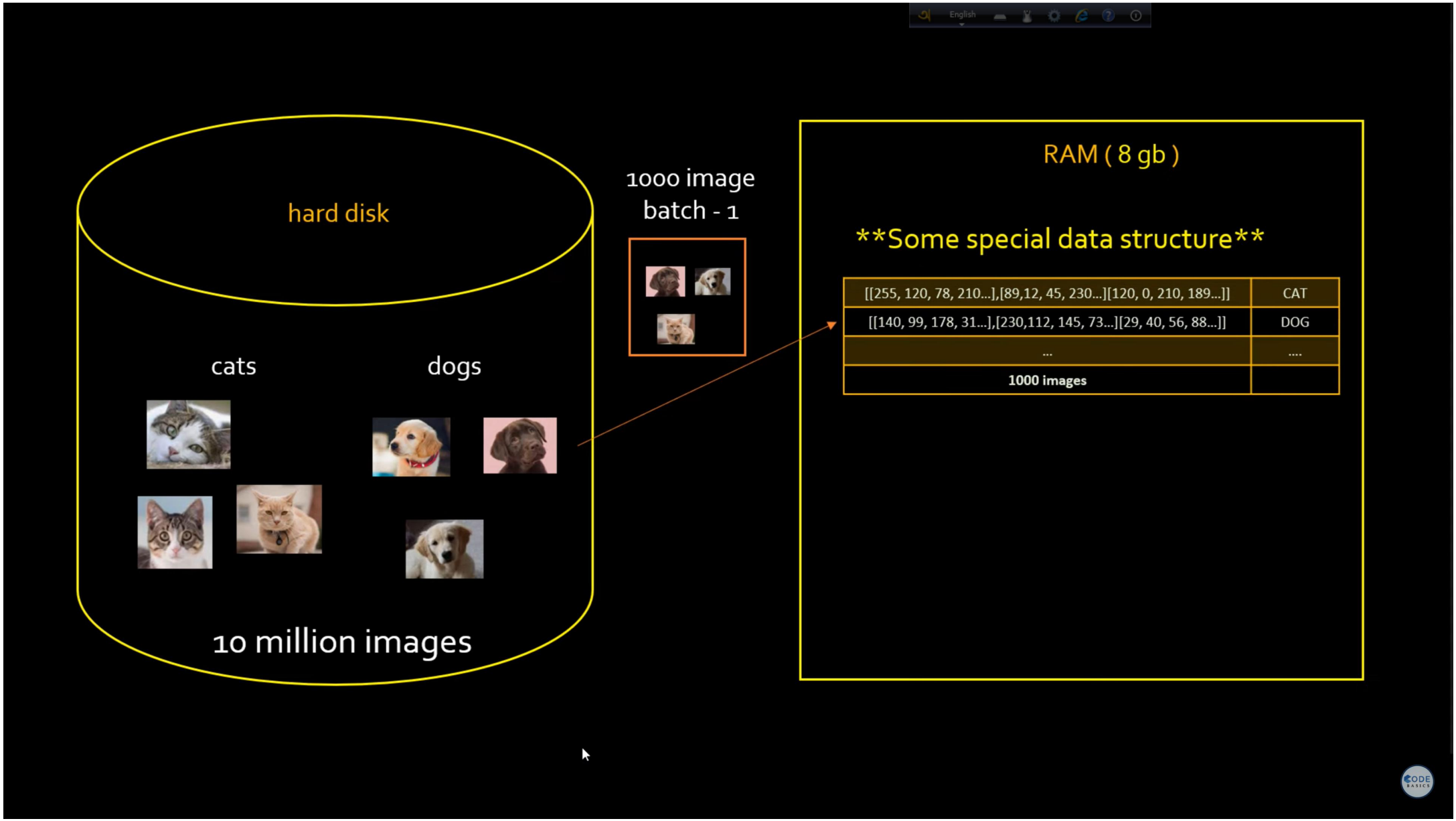
RAM (8 gb)

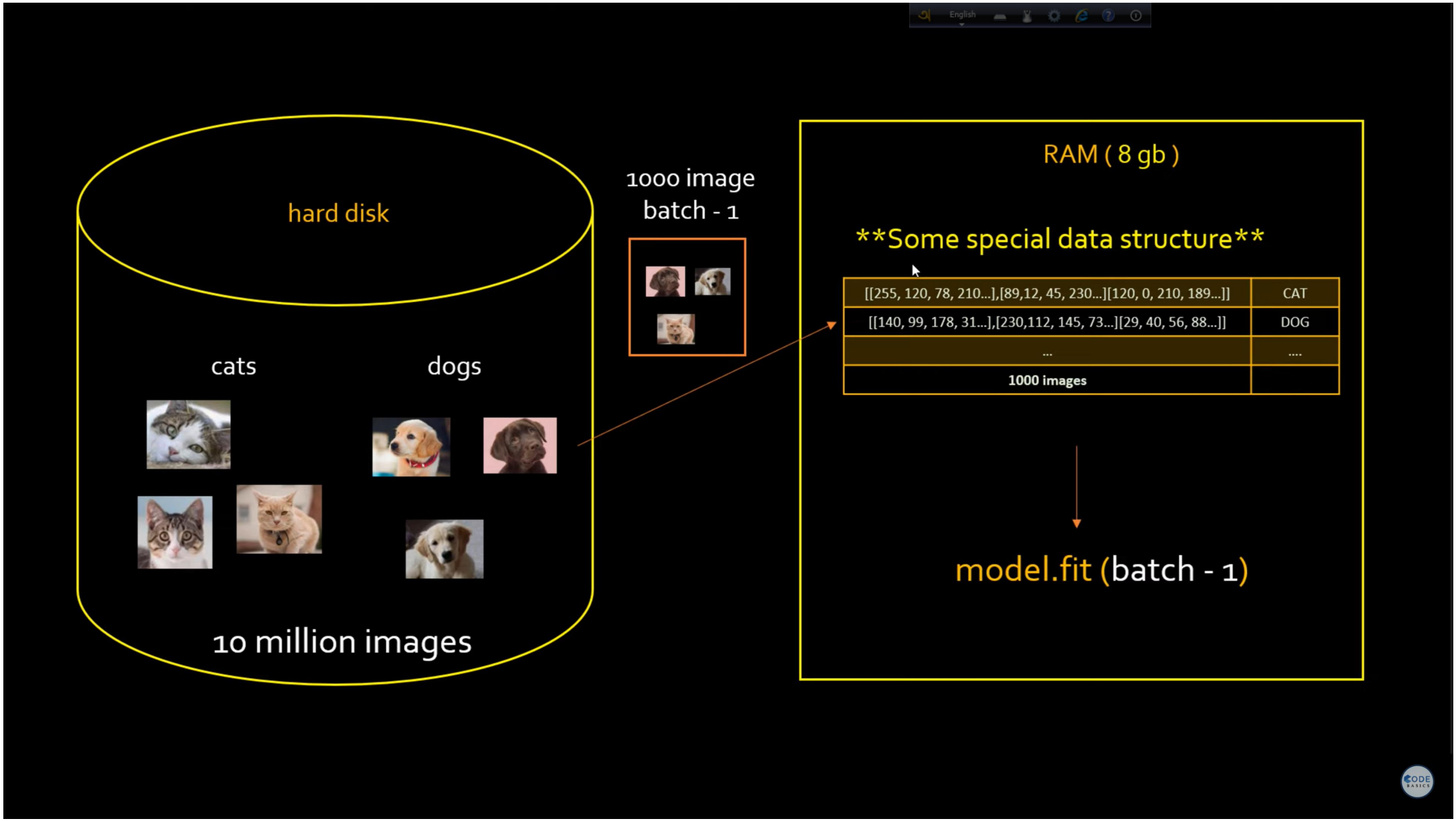

hard disk
cats
dogs
10 million images
RAM (8 gb)
TOO MUCH DATA
imgflip.com

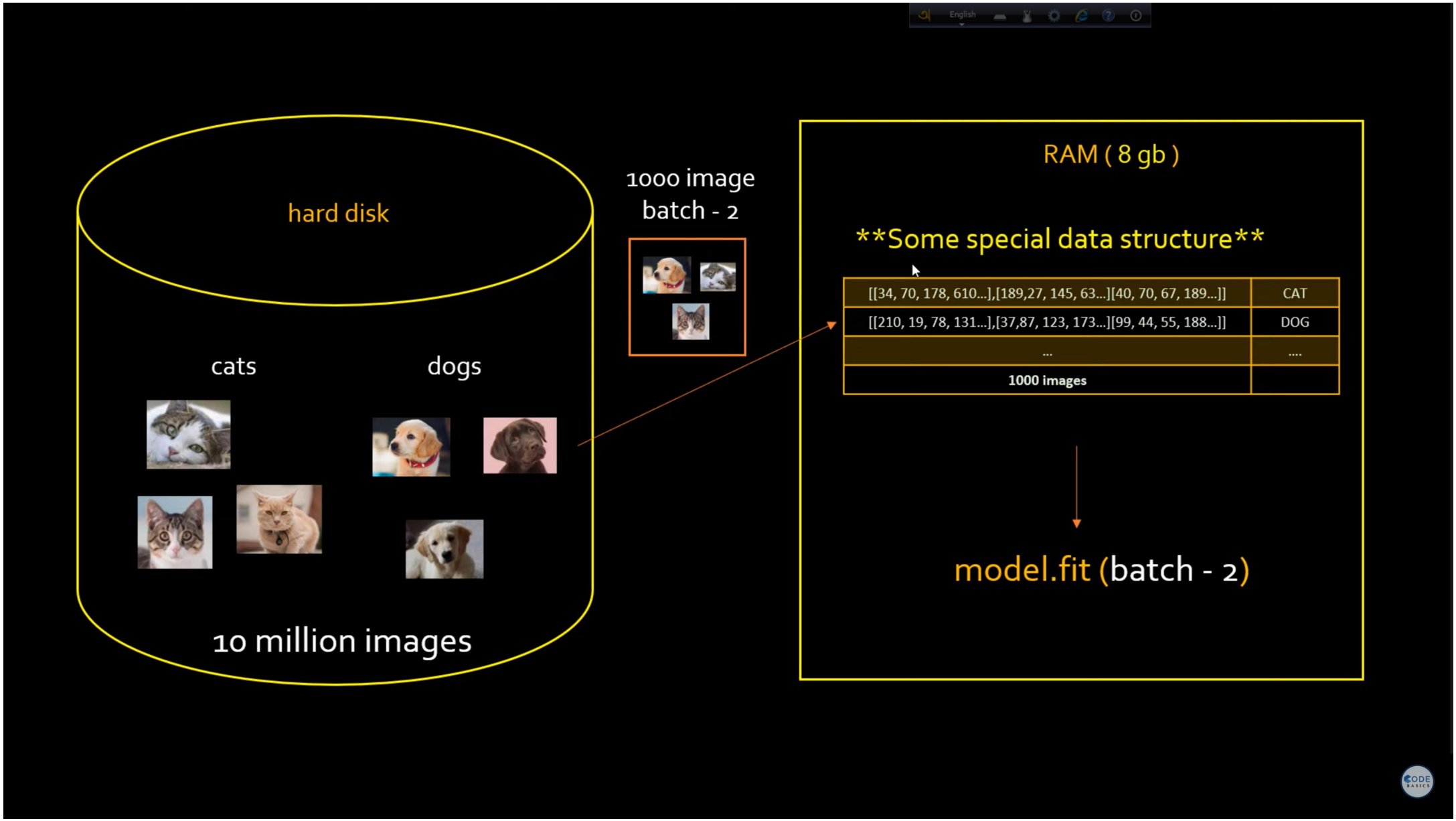

hard disk
cats
dogs
10 million images

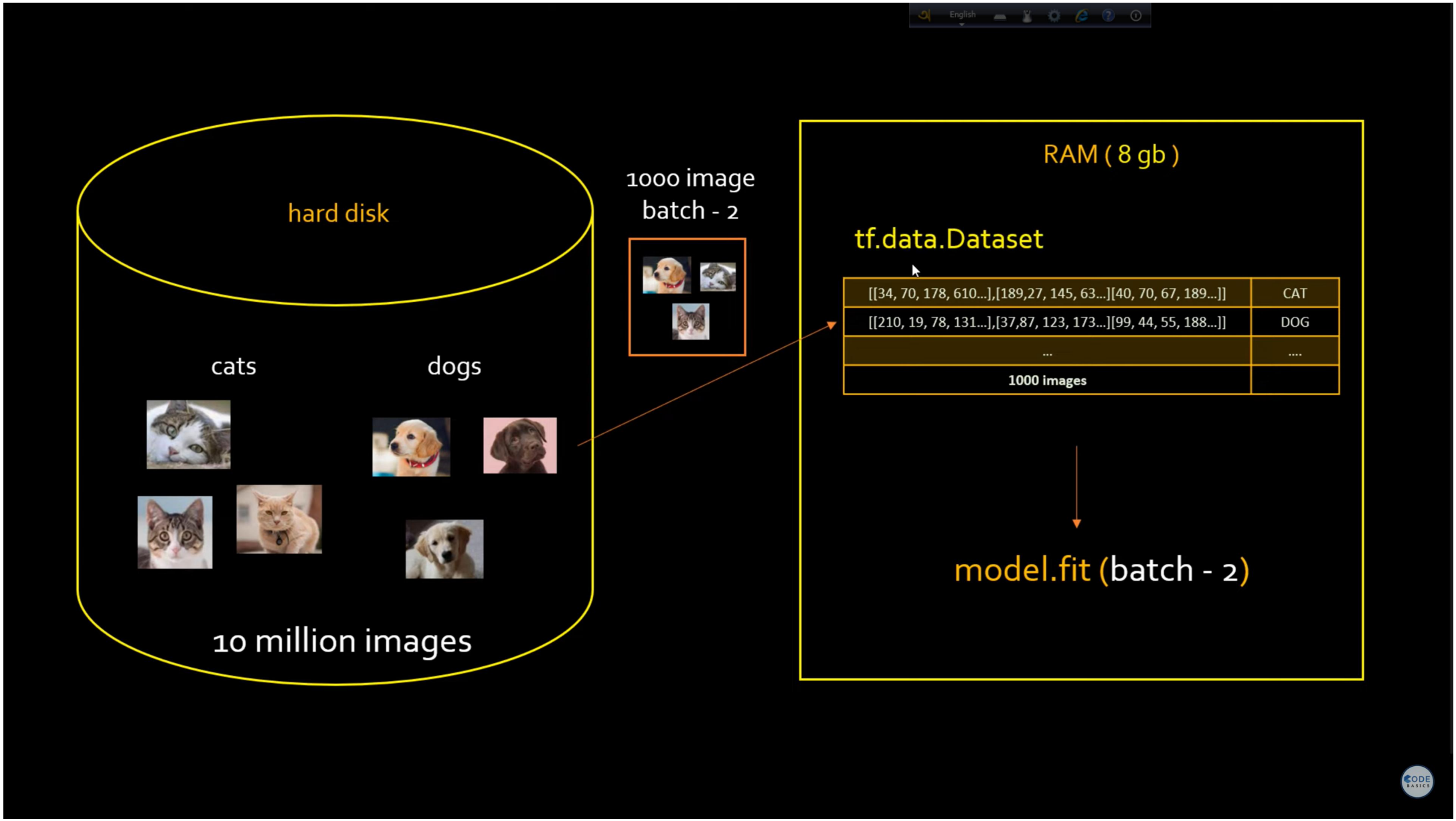

RAM (8 gb)

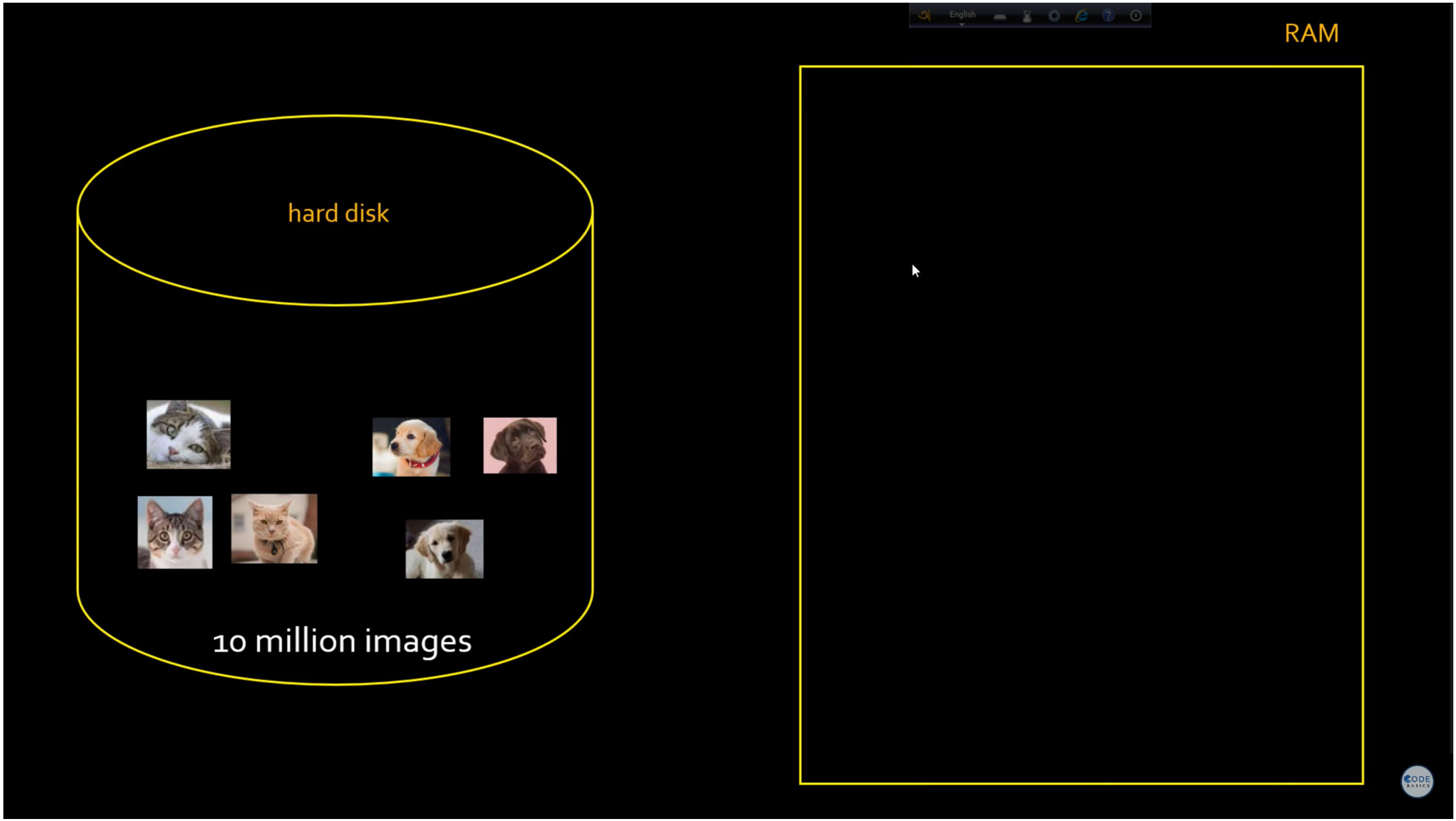

hard disk
cats
dogs
10 million images
1000 image
batch - 1
RAM (8 gb)

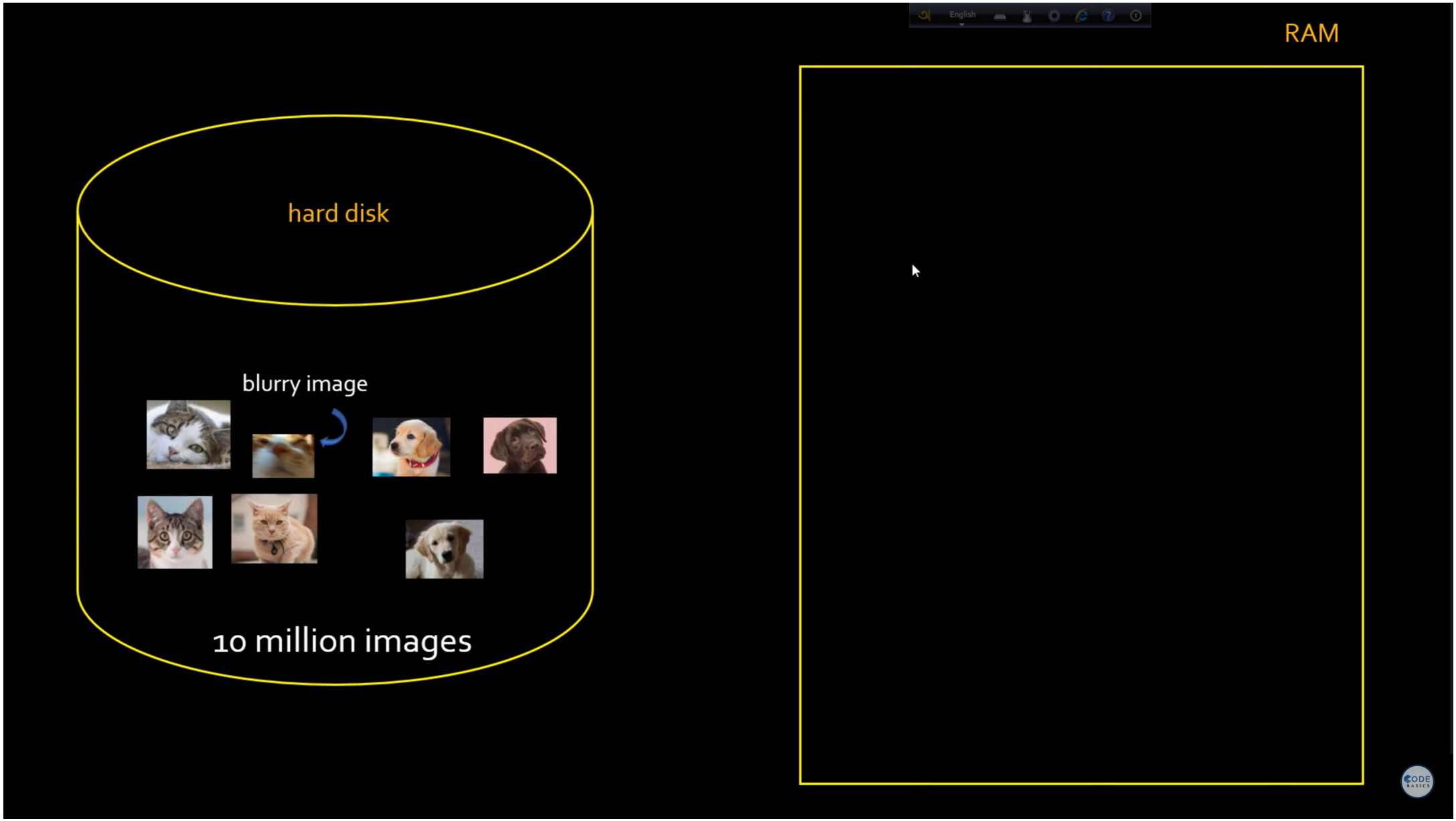

hard disk
cats
dogs
10 million images
1000 image batch - 1
RAM (8 gb)
Some special data structure
[[255, 120, 78, 210...],[89,12, 45, 230...][120, 0, 210, 189...]]
CAT
[[140, 99, 178, 31...],[230,112, 145, 73...][29, 40, 56, 88...]]
DOG
...
....
1000 images

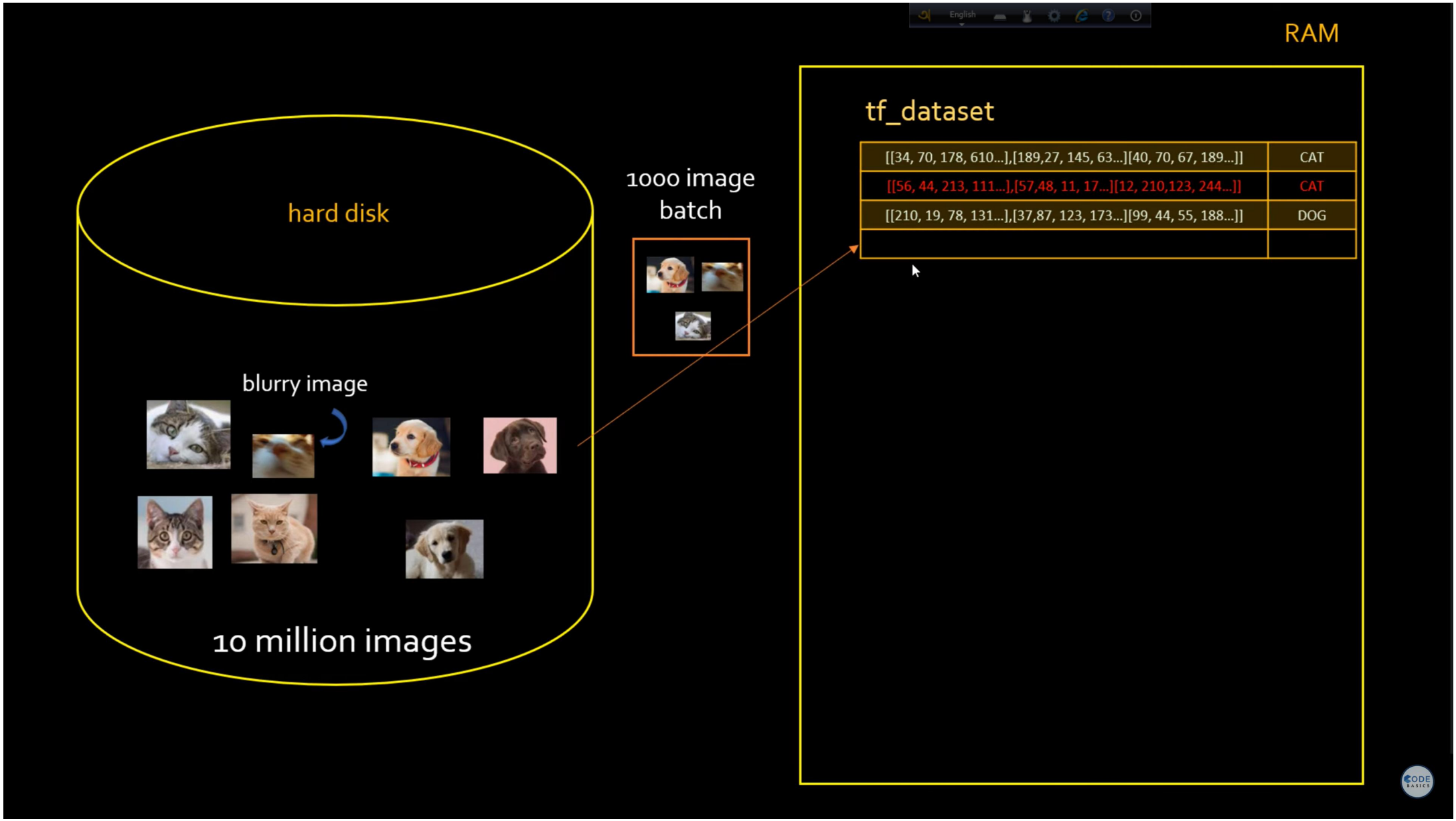

hard disk
cats
dogs
10 million images
1000 image batch - 1
RAM (8 gb)
Some special data structure
[[255, 120, 78, 210...],[89,12, 45, 230...][120, 0, 210, 189...]]
CAT
[[140, 99, 178, 31...],[230,112, 145, 73...][29, 40, 56, 88...]]
DOG
...
....
1000 images
model.fit (batch - 1)

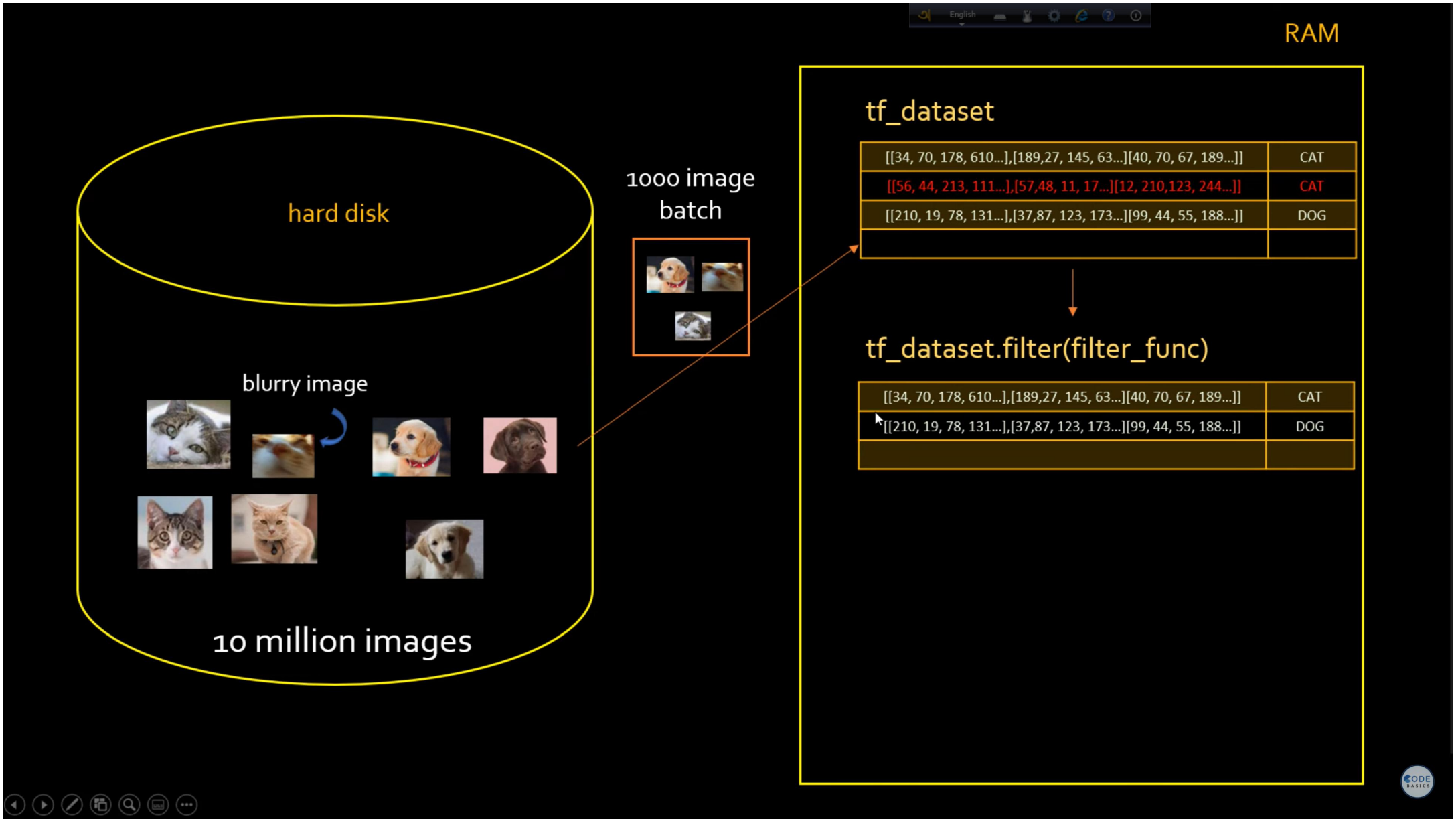

hard disk
cats
dogs
10 million images
1000 image batch - 2
RAM (8 gb)
Some special data structure
[[34, 70, 178, 610...],[189,27, 145, 63...][40, 70, 67, 189...]]
CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]]
DOG
...
....
1000 images
model.fit (batch - 2)

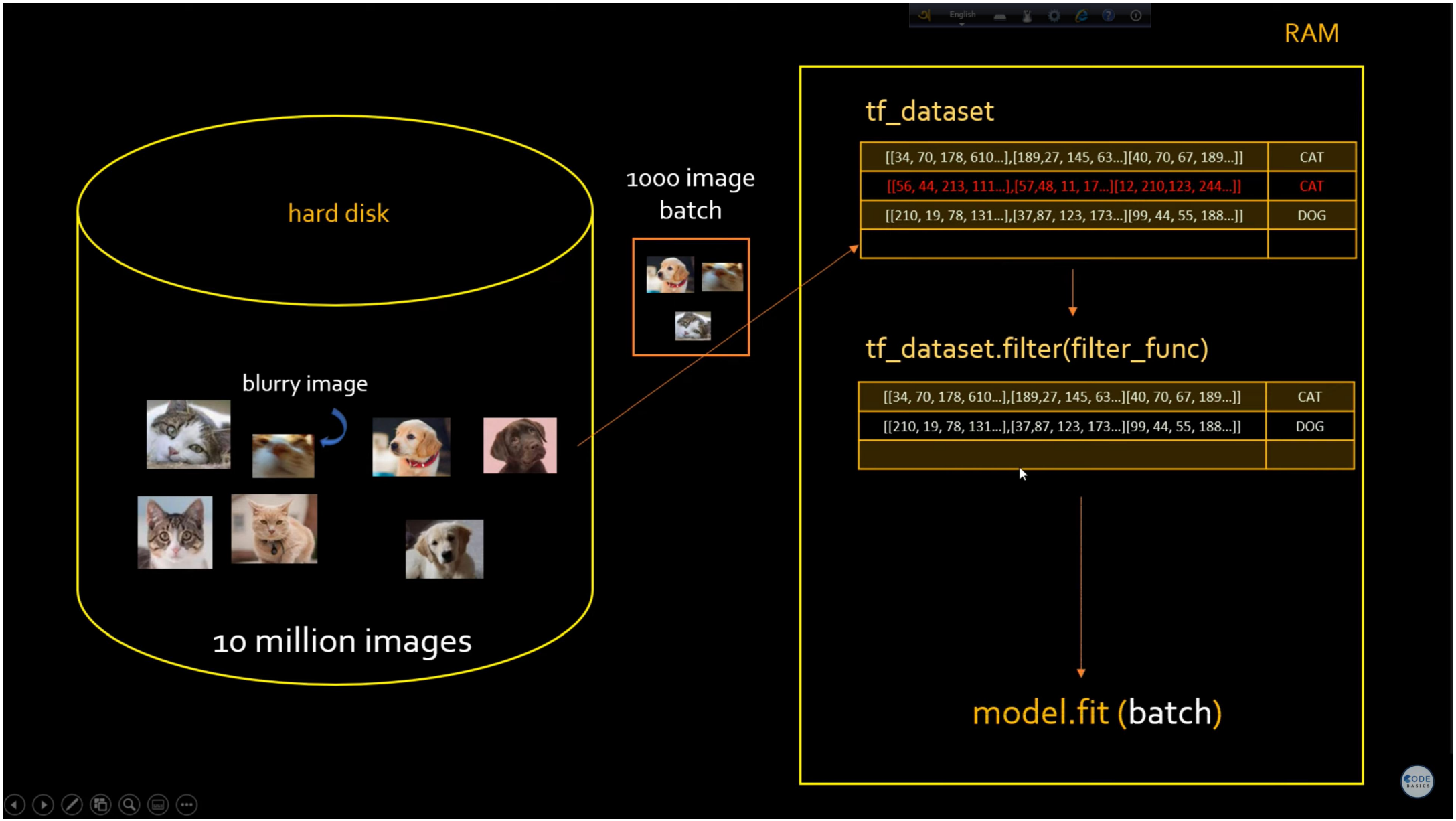

hard disk
cats
dogs
10 million images
1000 image
batch - 2
RAM (8 gb)
tf.data.Dataset
[[34, 70, 178, 610...],[189,27, 145, 63...][40, 70, 67, 189...]]
CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]]
DOG
...
....
1000 images
model.fit (batch - 2)

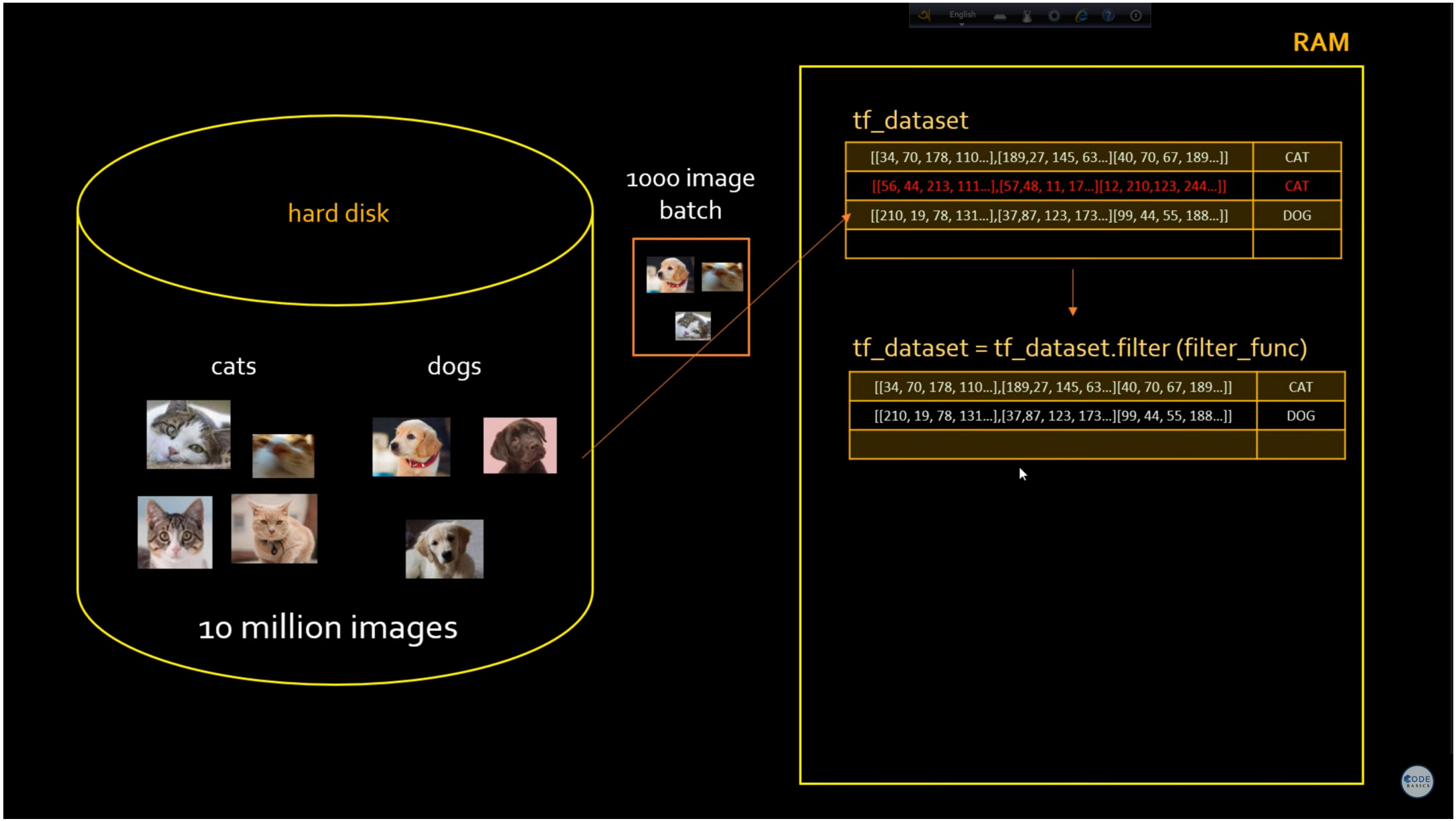

hard disk
10 million images
RAM

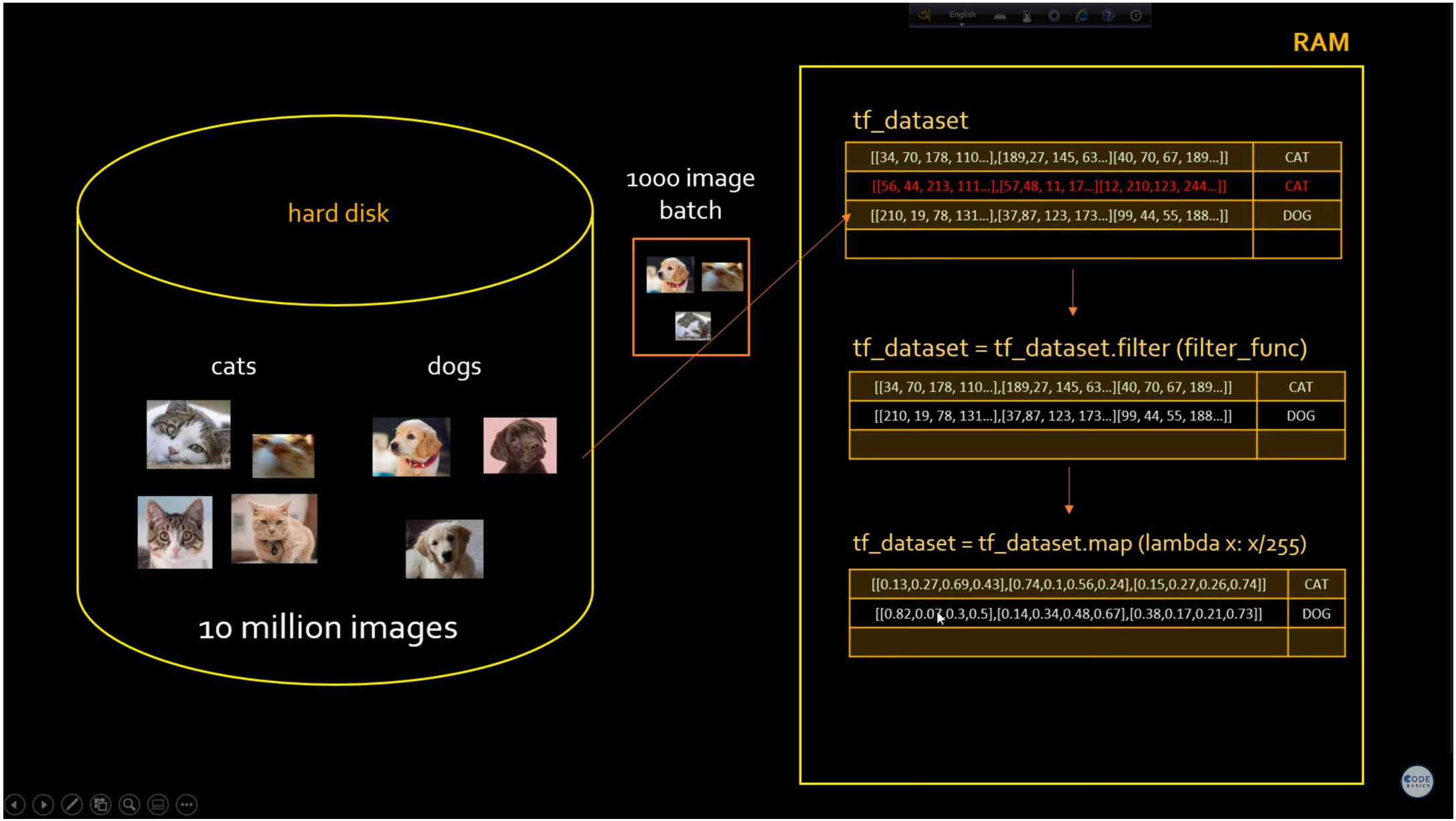

hard disk
blurry image
10 million images
RAM

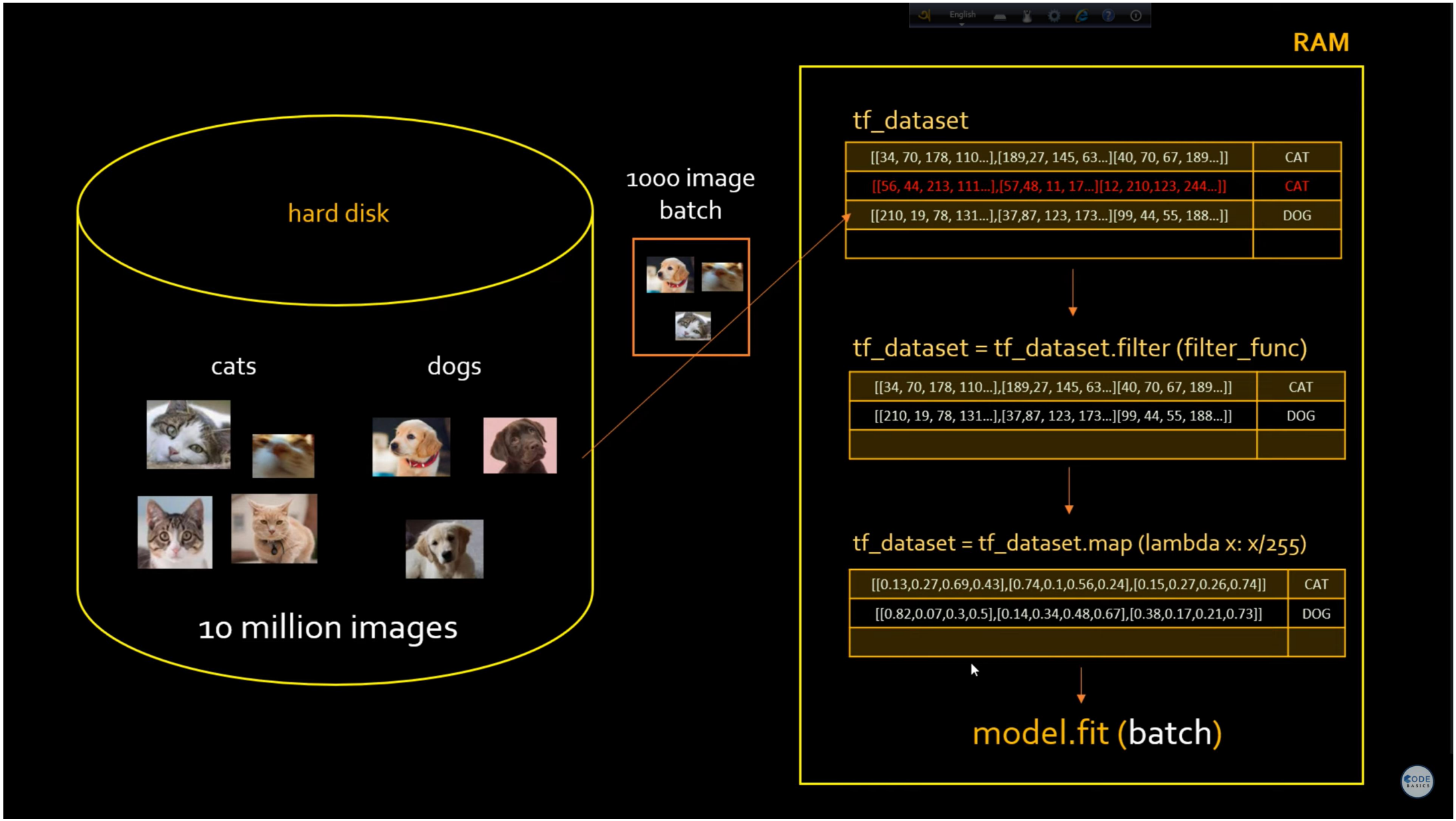

RAM
tf_dataset
[[34, 70, 178, 610...],[189,27, 145, 63...][40, 70, 67, 189...]]
CAT
[[56, 44, 213, 111...],[57,48, 11, 17...][12, 210,123, 244...]]
CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]]
DOG
1000 image batch
hard disk
blurry image
10 million images


RAM
hard disk
1000 image batch
blurry image
10 million images
tf_dataset
[[34, 70, 178, 610...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[56, 44, 213, 111...],[57,48, 11, 17...][12, 210,123, 244...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG
tf_dataset.filter(filter_func)
[[34, 70, 178, 610...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG


RAM
hard disk
1000 image batch
blurry image
10 million images
tf_dataset
[[34, 70, 178, 610...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[56, 44, 213, 111...],[57,48, 11, 17...][12, 210,123, 244...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG
tf_dataset.filter(filter_func)
[[34, 70, 178, 610...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG
model.fit (batch)

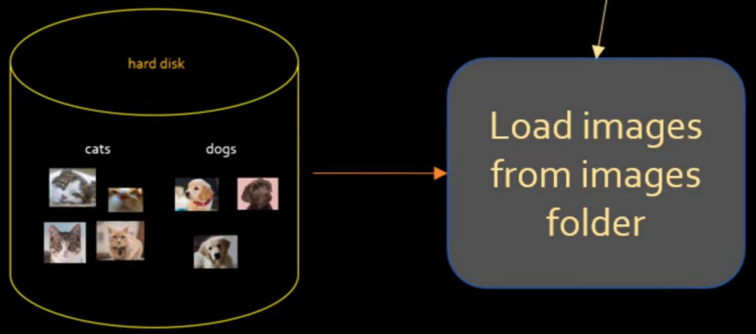

RAM
hard disk
cats
dogs
10 million images
1000 image batch
tf_dataset
[[34, 70, 178, 110...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[56, 44, 213, 111...],[57,48, 11, 17...][12, 210,123, 244...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG
tf_dataset = tf_dataset.filter (filter_func)
[[34, 70, 178, 110...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG


RAM
hard disk
cats
dogs
10 million images
1000 image batch
tf_dataset
[[34, 70, 178, 110...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[56, 44, 213, 111...],[57,48, 11, 17...][12, 210,123, 244...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG
tf_dataset = tf_dataset.filter (filter_func)
[[34, 70, 178, 110...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG
tf_dataset = tf_dataset.map (lambda x: x/255)
[[0.13,0.27,0.69,0.43],[0.74,0.1,0.56,0.24],[0.15,0.27,0.26,0.74]] CAT
[[0.82,0.07,0.3,0.5],[0.14,0.34,0.48,0.67],[0.38,0.17,0.21,0.73]] DOG


RAM
hard disk
cats
dogs
10 million images
1000 image batch
tf_dataset
[[34, 70, 178, 110...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[56, 44, 213, 111...],[57,48, 11, 17...][12, 210,123, 244...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG
tf_dataset = tf_dataset.filter (filter_func)
[[34, 70, 178, 110...],[189,27, 145, 63...][40, 70, 67, 189...]] CAT
[[210, 19, 78, 131...],[37,87, 123, 173...][99, 44, 55, 188...]] DOG
tf_dataset = tf_dataset.map (lambda x: x/255)
[[0.13,0.27,0.69,0.43],[0.74,0.1,0.56,0.24],[0.15,0.27,0.26,0.74]] CAT
[[0.82,0.07,0.3,0.5],[0.14,0.34,0.48,0.67],[0.38,0.17,0.21,0.73]] DOG
model.fit (batch)

```
tf_dataset = tf.data.Dataset.list_files('images/*').map(process_img).filter(filter_func).map (lambda x: x/255)
```

```
tf_dataset = tf.data.Dataset.list_files('images/*').map(process_img).filter(filter_func).map (lambda x: x/255)
```



Load images from images folder

```
tf_dataset = tf.data.Dataset.list_files('images/*').map(process_img).filter(filter_func).map (lambda x: x/255)
```

Load images from images folder

Convert image content to numpy array. Extract label from folder

```
tf_dataset = tf.data.Dataset.list_files('images/*').map(process_img).filter(filter_func).map (lambda x: x/255)
```

Load images from images folder

Convert image content to numpy array. Extract label from folder

Filter Blurred Images

```
tf_dataset = tf.data.Dataset.list_files('images/*').map(process_img).filter(filter_func).map (lambda x: x/255)
```

Load images from images folder

Convert image content to numpy array. Extract label from folder

Filter Blurred Images

Scaling

tf_dataset

Step 1: Build Data Pipeline

```
tf_dataset = tf.data.Dataset.list_files('images/*').map(process_img).filter(filter_func).map (lambda x: x/255)
```

Load images from images folder

Convert image content to numpy array. Extract label from folder

Filter Blurred Images

Scaling

tf_dataset

Step 1: Build Data Pipeline

```
tf_dataset = tf.data.Dataset.list_files('images/*').map(process_img).filter(filter_func).map (lambda x: x/255)
```

Load images from images folder

Convert image content to numpy array. Extract label from folder

Filter Blurred Images

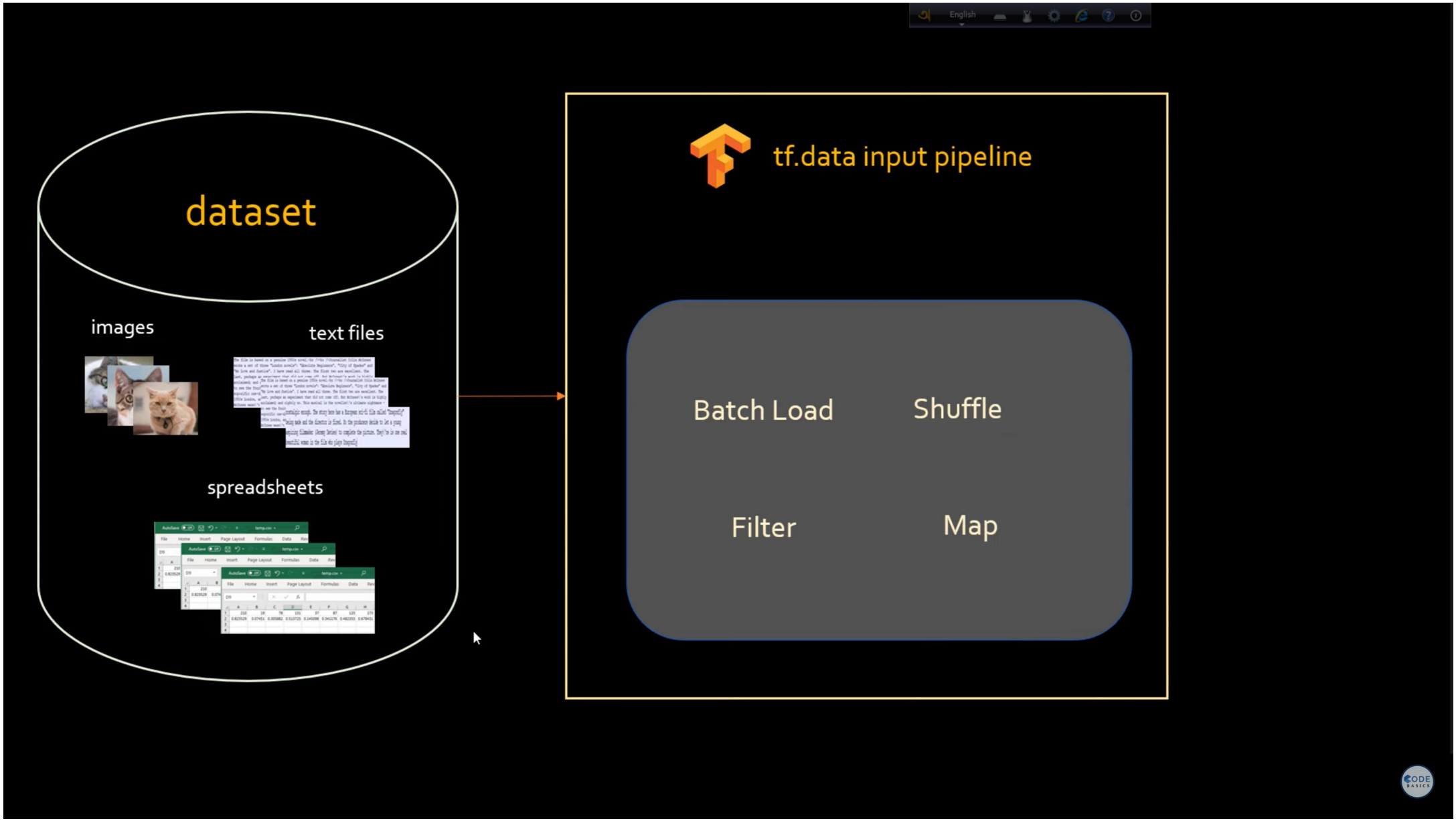
Scaling

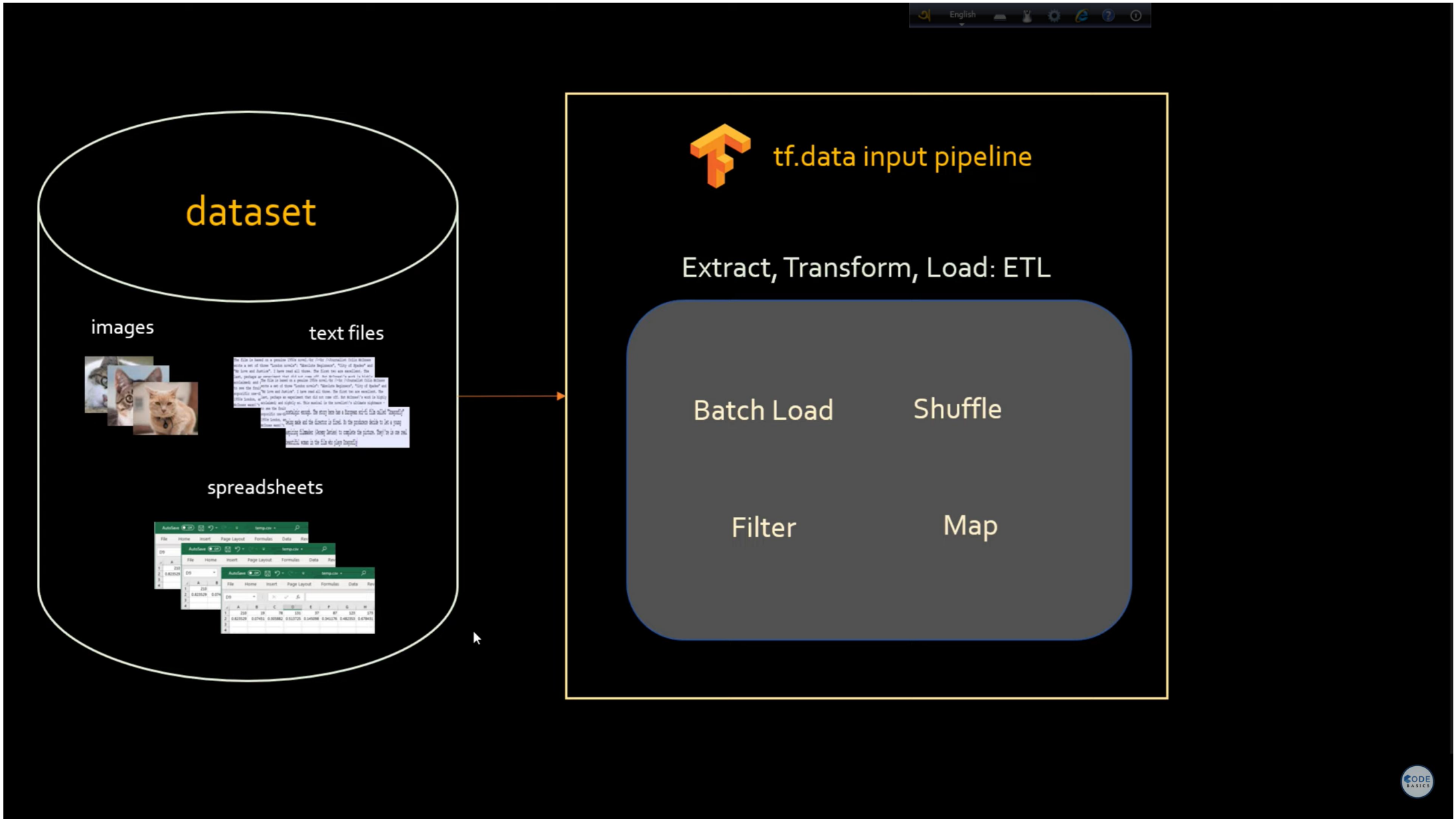
tf_dataset

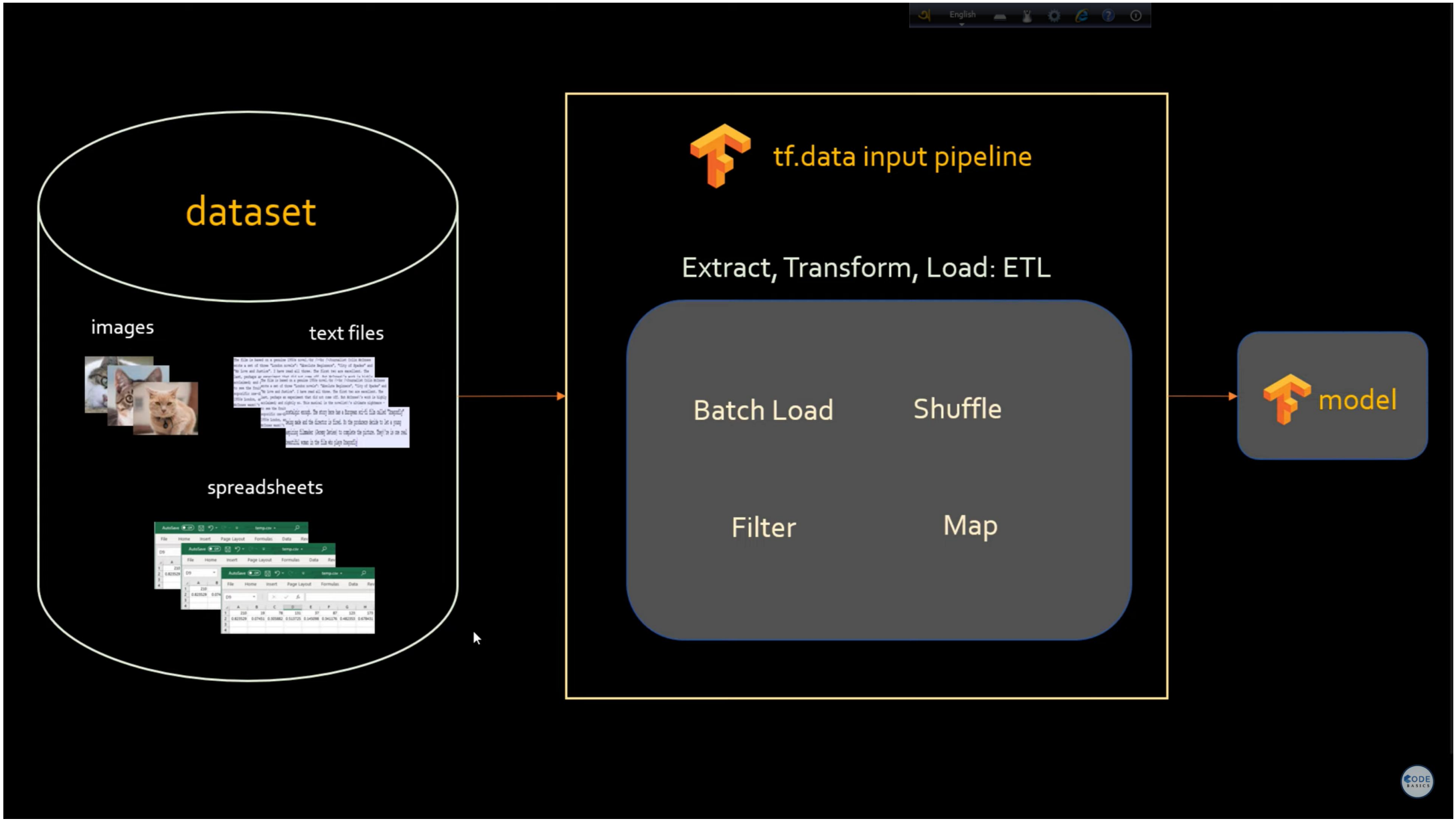
Step 2: Train the model

```
model.fit(tf_dataset)
```


dataset
images
text files
spreadsheets


dataset
images
text files
spreadsheets
tf.data input pipeline
Batch Load
Shuffle
Filter
Map


dataset
images
text files
spreadsheets
tf.data input pipeline
Extract, Transform, Load: ETL
Batch Load
Shuffle
Filter
Map


dataset
images
text files
spreadsheets
tf.data input pipeline
Extract, Transform, Load: ETL
Batch Load
Shuffle
Filter
Map
model

Tensorflow input pipeline benefits

Handle huge datasets by streaming them from disk using batching

Apply transformations to make dataset ready for model training