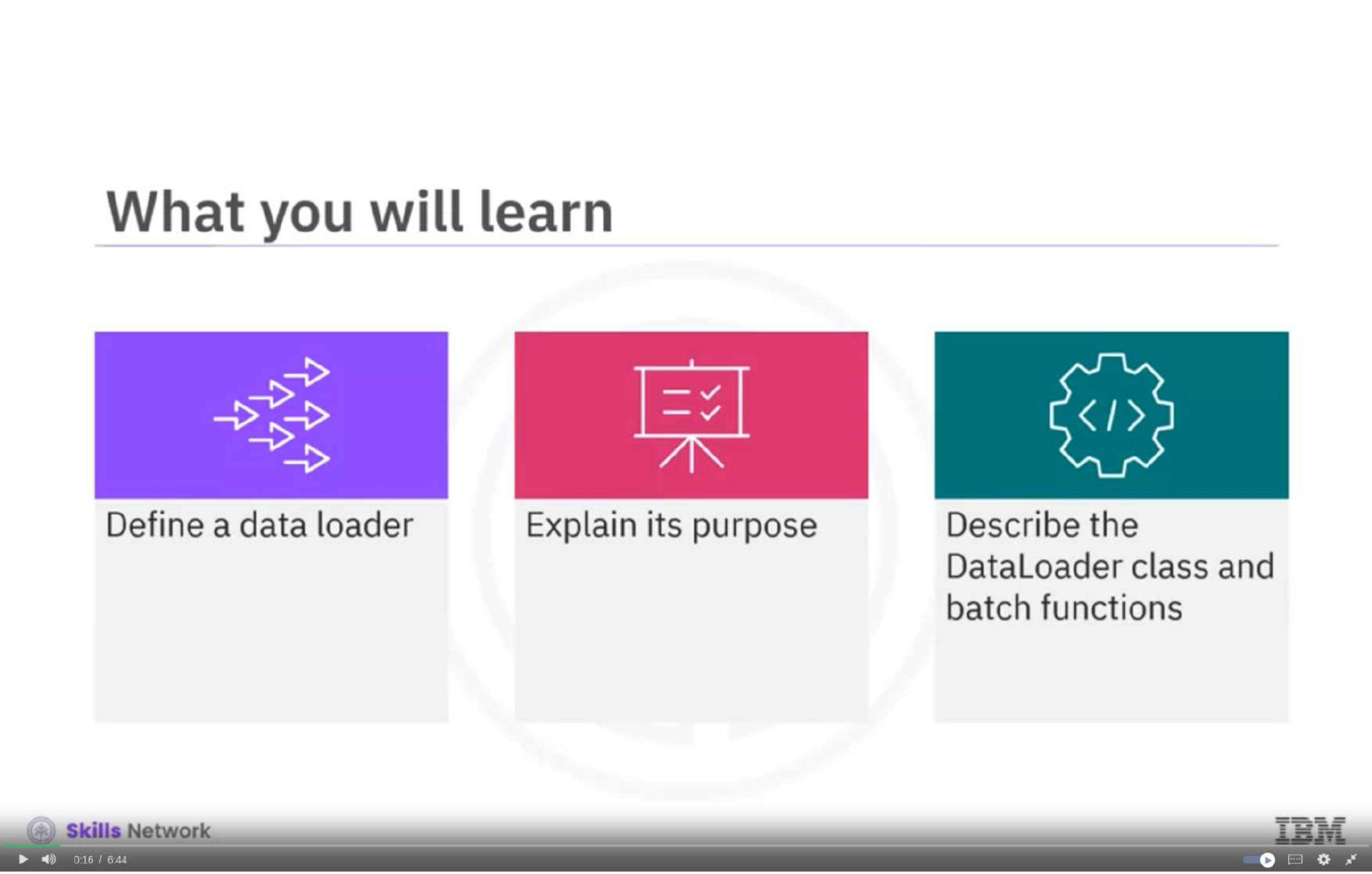
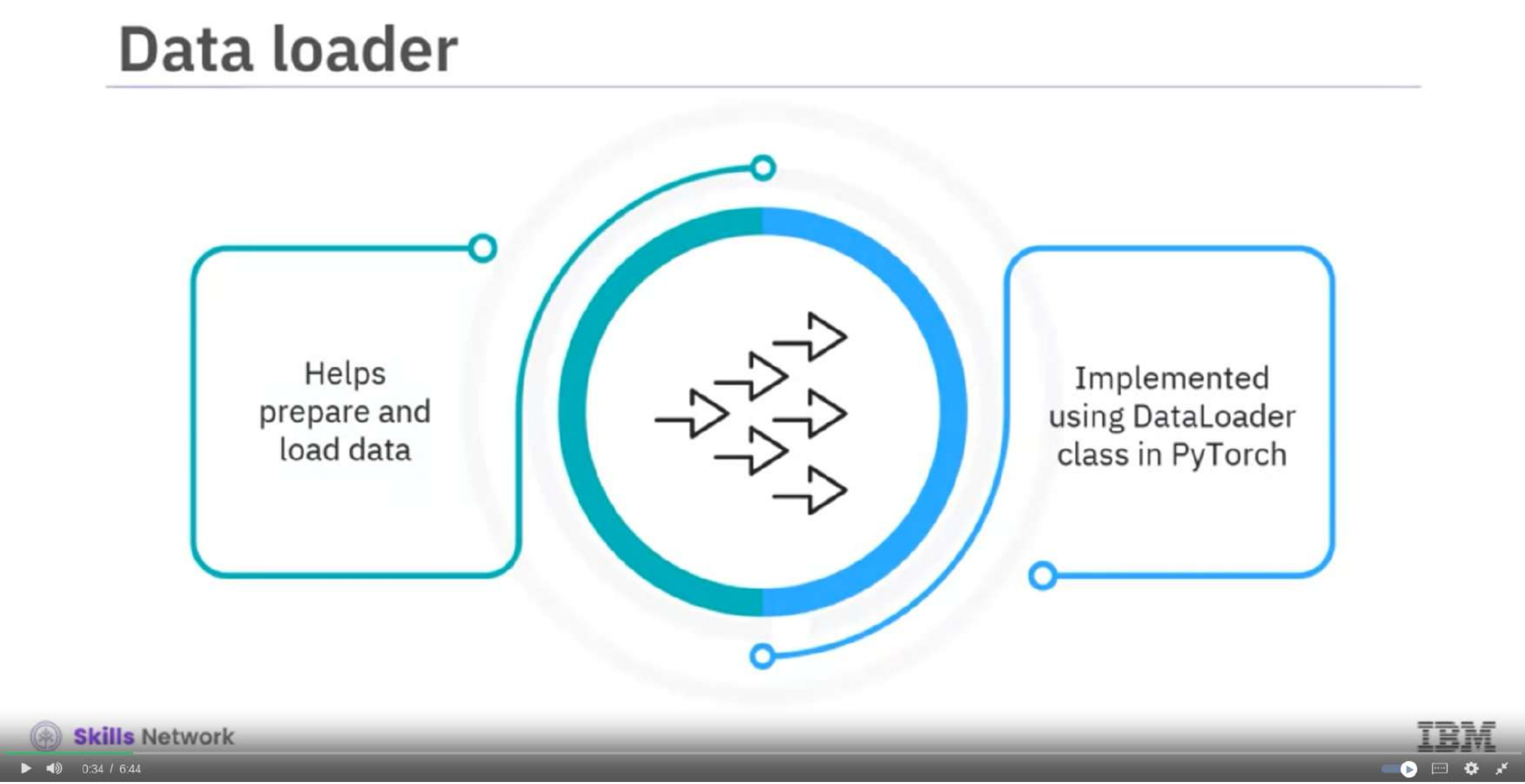


What you will learn

Define a data loader

Explain its purpose

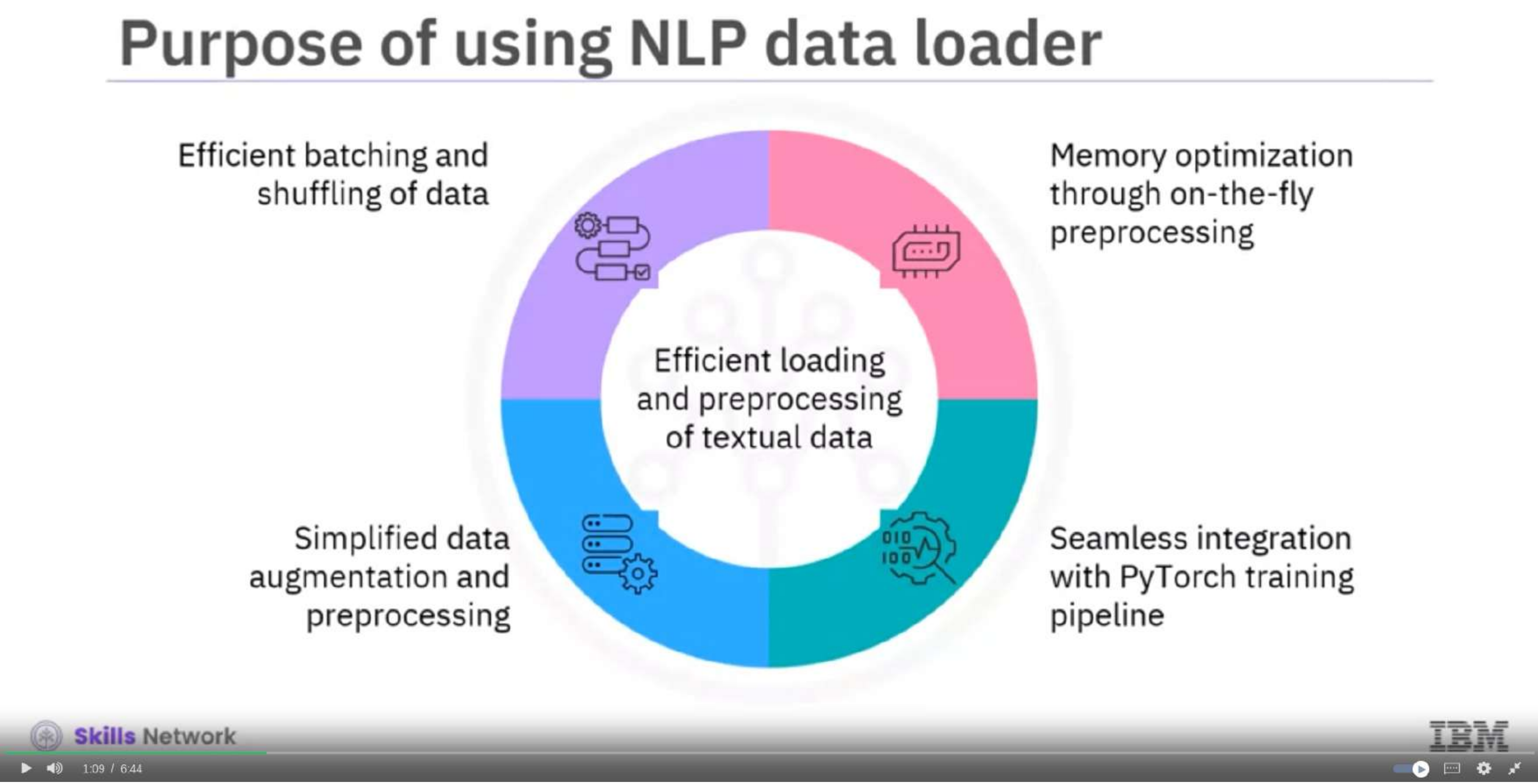
Describe the DataLoader class and batch functions



Data loader

Helps prepare and load data

Implemented using DataLoader class in PyTorch



Purpose of using NLP data loader

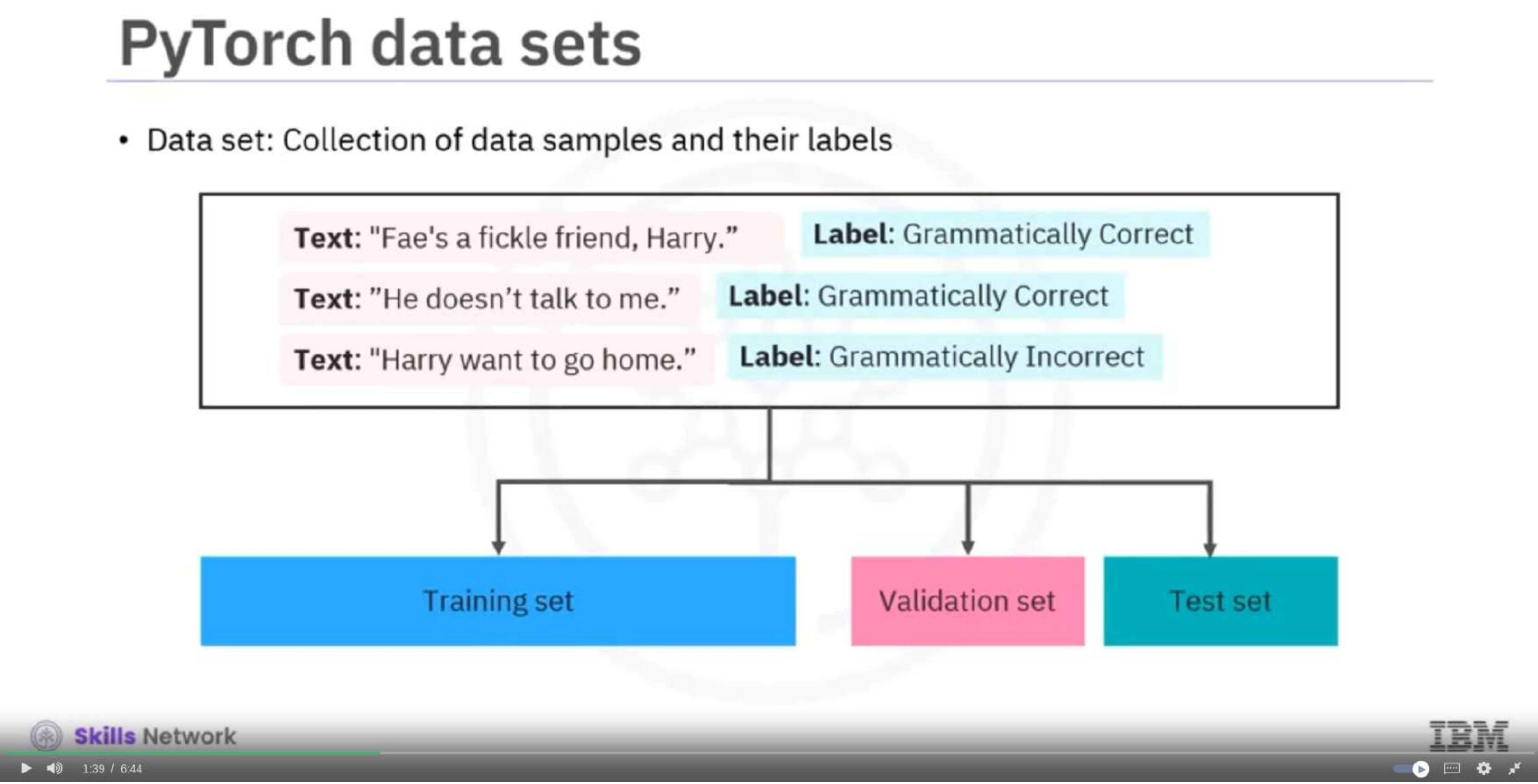
Efficient loading and preprocessing of textual data

Efficient batching and shuffling of data

Memory optimization through on-the-fly preprocessing

Simplified data augmentation and preprocessing

Seamless integration with PyTorch training pipeline



PyTorch data sets

- Data set: Collection of data samples and their labels

Text: "Fae's a fickle friend, Harry." **Label**: Grammatically Correct

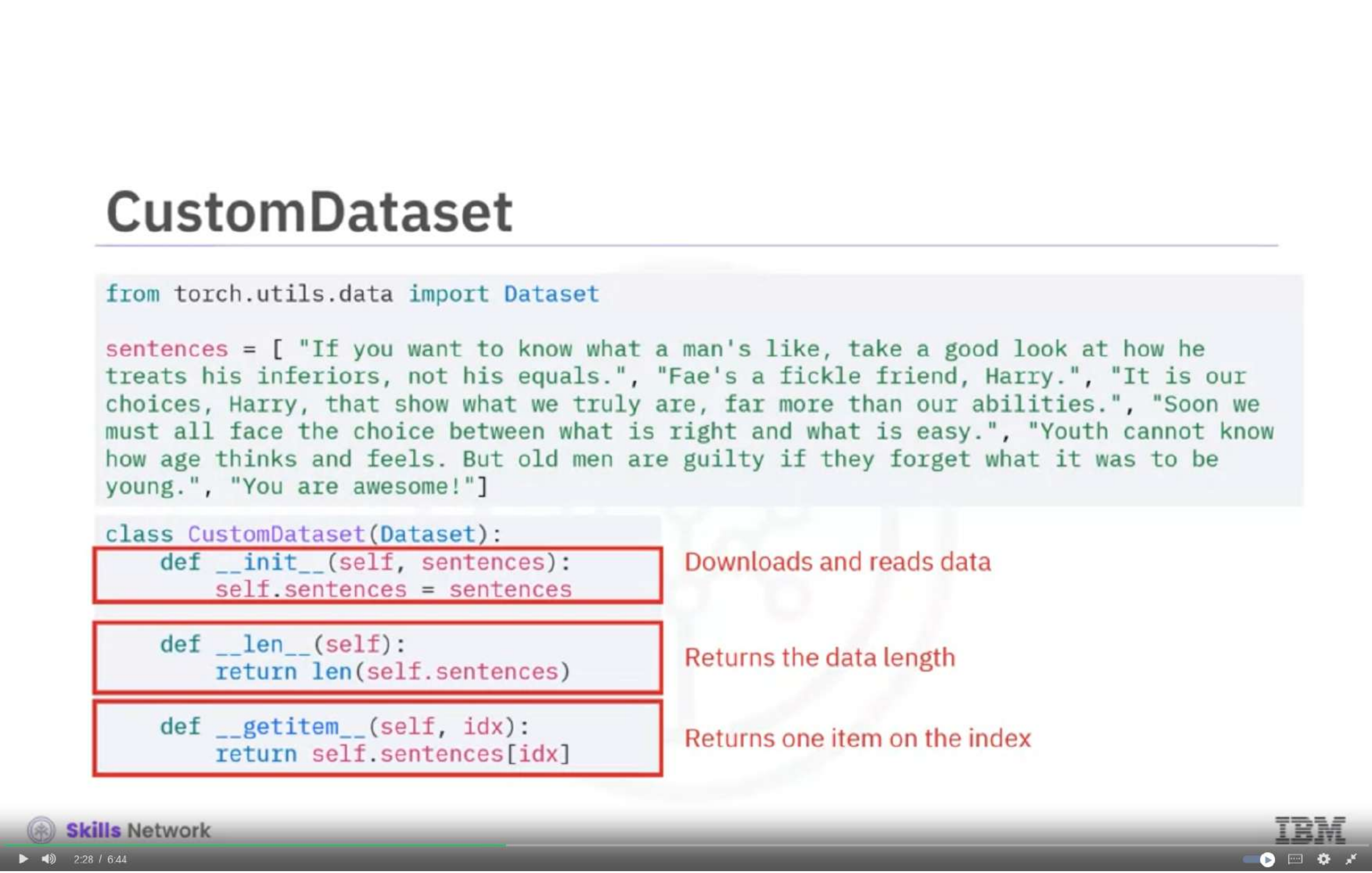
Text: "He doesn't talk to me." **Label**: Grammatically Correct

Text: "Harry want to go home." **Label**: Grammatically Incorrect

Training set

Validation set

Test set



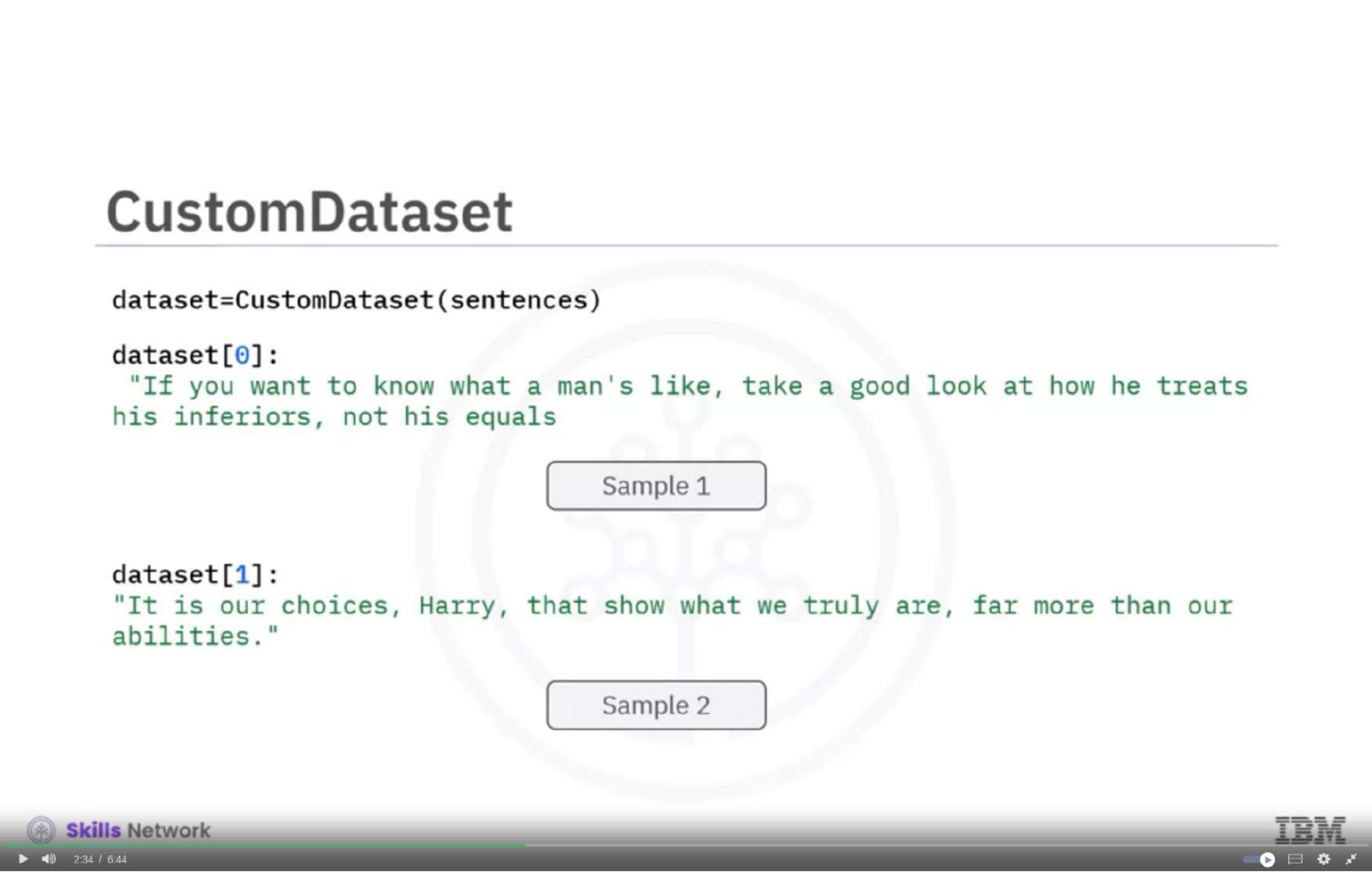
CustomDataset

```python
from torch.utils.data import Dataset

sentences = [ "If you want to know what a man's like, take a good look at how he
treats his inferiors, not his equals.", "Fae's a fickle friend, Harry.", "It is our
choices, Harry, that show what we truly are, far more than our abilities.", "Soon we
must all face the choice between what is right and what is easy.", "Youth cannot know
how age thinks and feels. But old men are guilty if they forget what it was to be
young.", "You are awesome!"]

class CustomDataset(Dataset):
    def __init__(self, sentences):
        self.sentences = sentences

    def __len__(self):
        return len(self.sentences)

    def __getitem__(self, idx):
        return self.sentences[idx]
```

Downloads and reads data

Returns the data length

Returns one item on the index



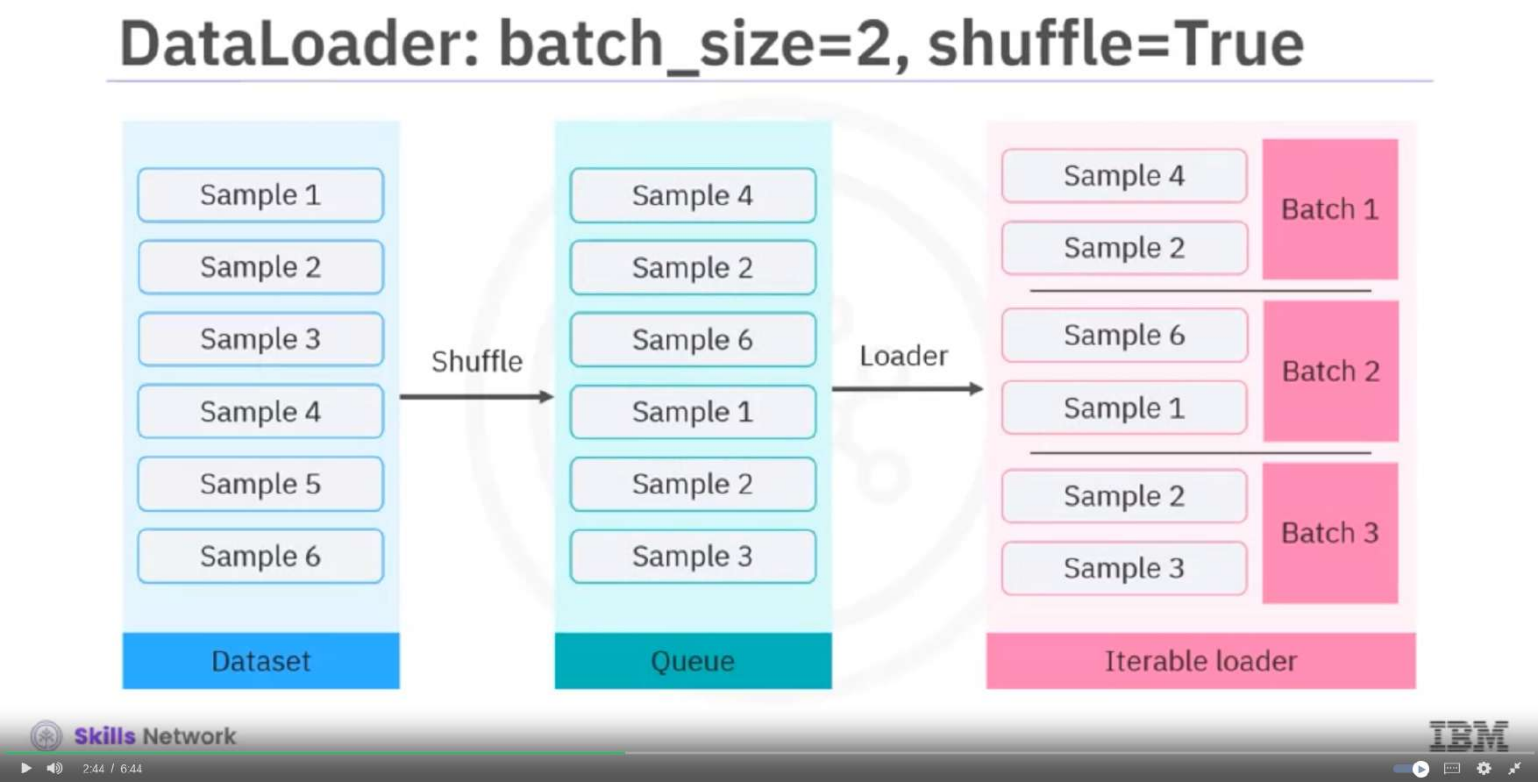
CustomDataset

```
dataset=CustomDataset(sentences)

dataset[0]:
 "If you want to know what a man's like, take a good look at how he treats
his inferiors, not his equals
```

Sample 1

```
dataset[1]:
"It is our choices, Harry, that show what we truly are, far more than our
abilities."
```

Sample 2



DataLoader: batch_size=2, shuffle=True

Dataset

Sample 1

Sample 2

Sample 3

Sample 4

Sample 5

Sample 6

Shuffle

Queue

Sample 4

Sample 2

Sample 6

Sample 1

Sample 2

Sample 3

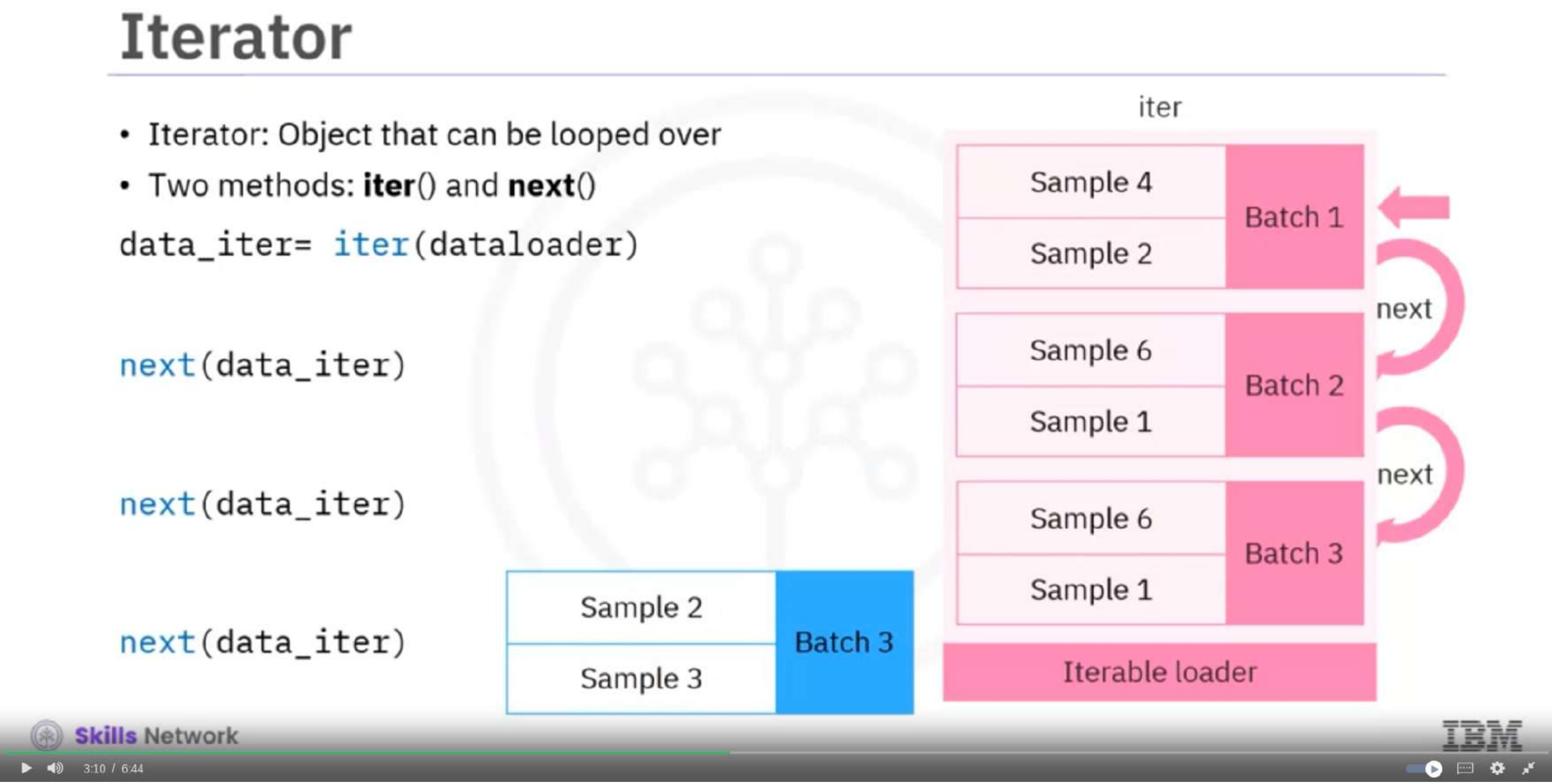
Loader

Iterable loader

Batch 1: Sample 4, Sample 2

Batch 2: Sample 6, Sample 1

Batch 3: Sample 2, Sample 3



Iterator

- Iterator: Object that can be looped over
- Two methods: **iter**() and **next**()

```
data_iter= iter(dataloader)

next(data_iter)

next(data_iter)

next(data_iter)
```

Sample 2

Sample 3

Batch 3

iter

Sample 4

Sample 2

Batch 1

next

Sample 6

Sample 1

Batch 2

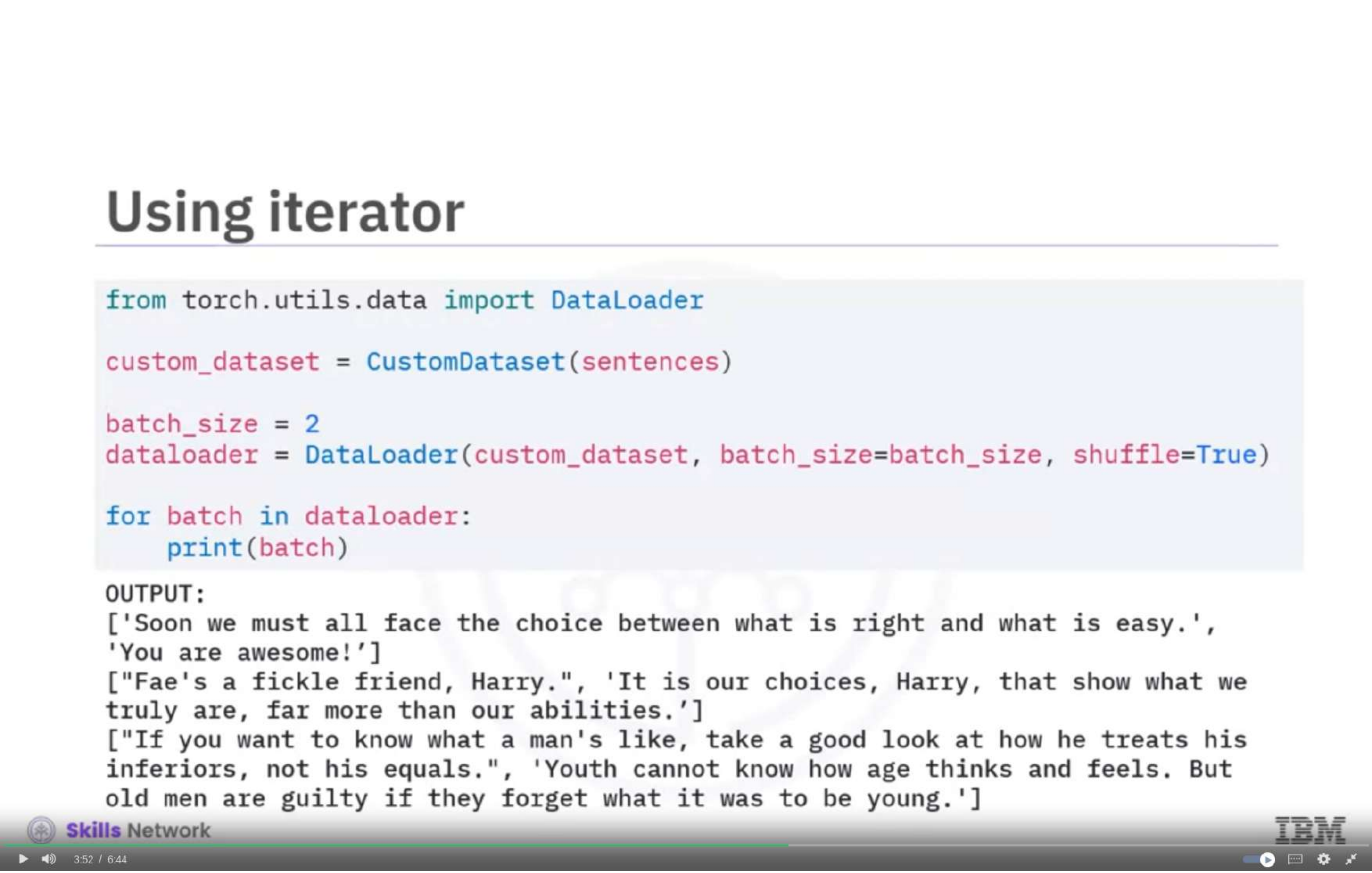
next

Sample 6

Sample 1

Batch 3

Iterable loader

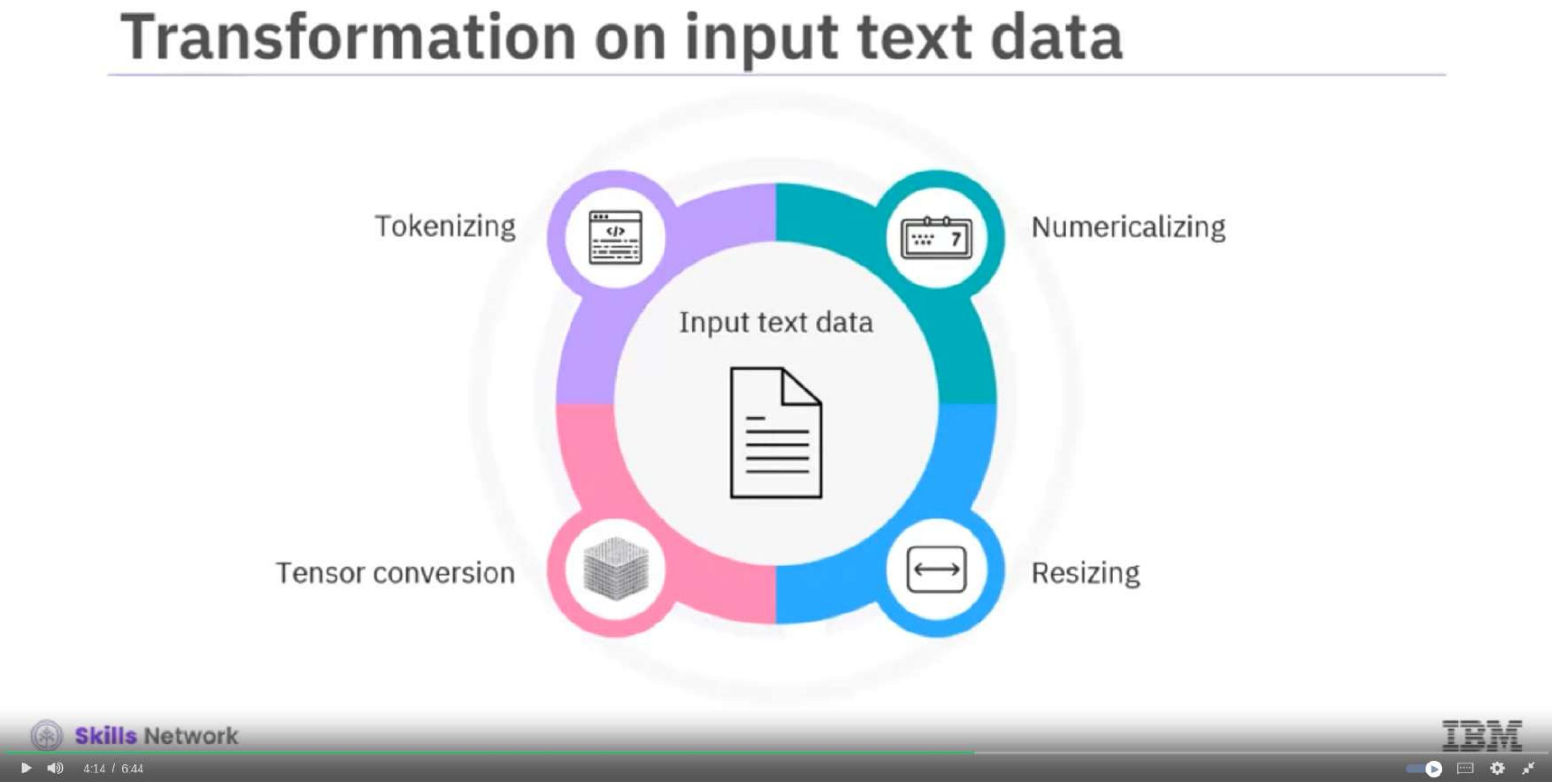


Using iterator

```python
from torch.utils.data import DataLoader

custom_dataset = CustomDataset(sentences)

batch_size = 2
dataloader = DataLoader(custom_dataset, batch_size=batch_size, shuffle=True)

for batch in dataloader:
    print(batch)
```

OUTPUT:

```
['Soon we must all face the choice between what is right and what is easy.',
'You are awesome!’]
["Fae's a fickle friend, Harry.", 'It is our choices, Harry, that show what we
truly are, far more than our abilities.’]
["If you want to know what a man's like, take a good look at how he treats his
inferiors, not his equals.", 'Youth cannot know how age thinks and feels. But
old men are guilty if they forget what it was to be young.']
```



Transformation on input text data

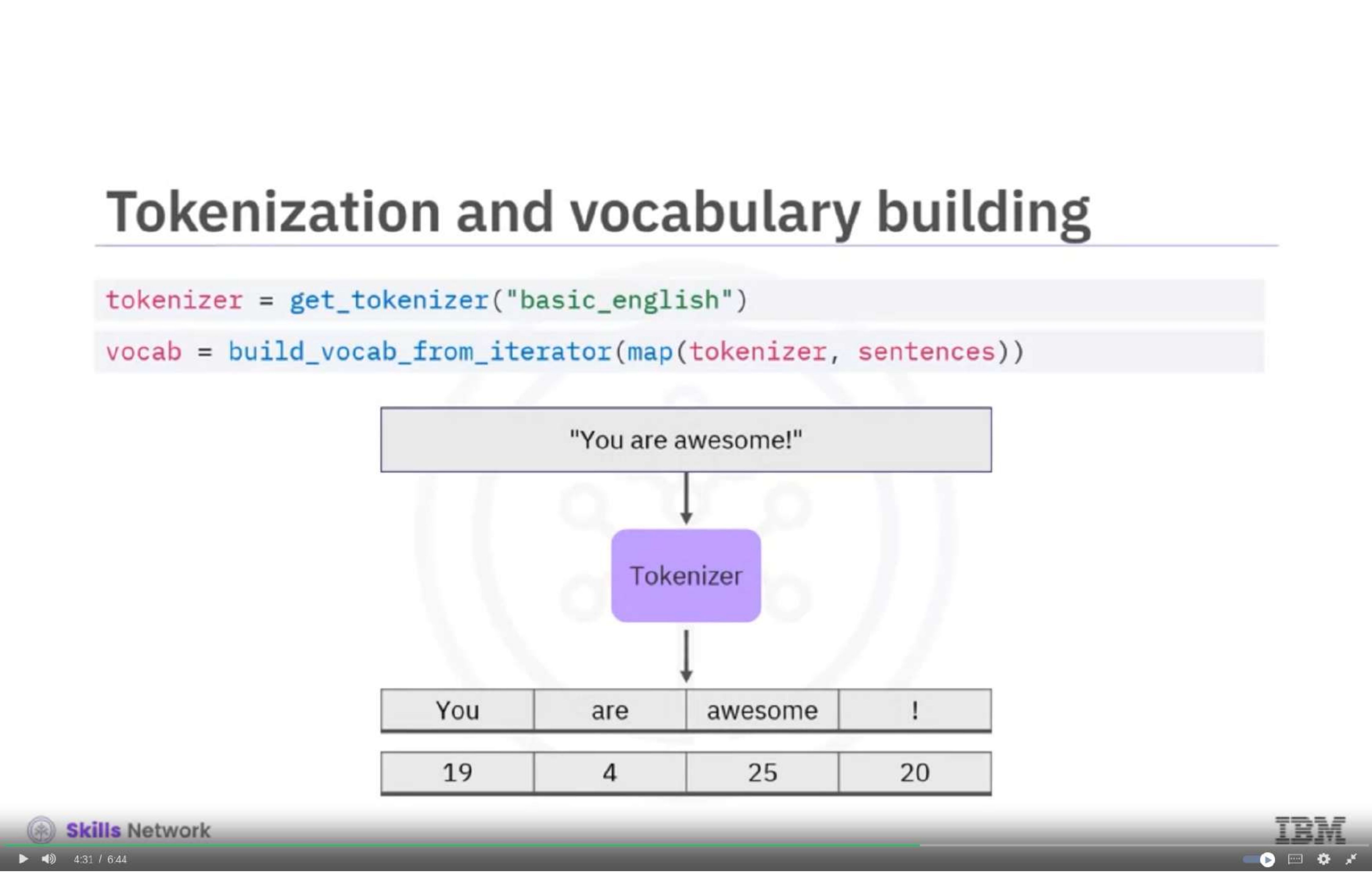
Input text data

Tokenizing

Numericalizing

Resizing

Tensor conversion



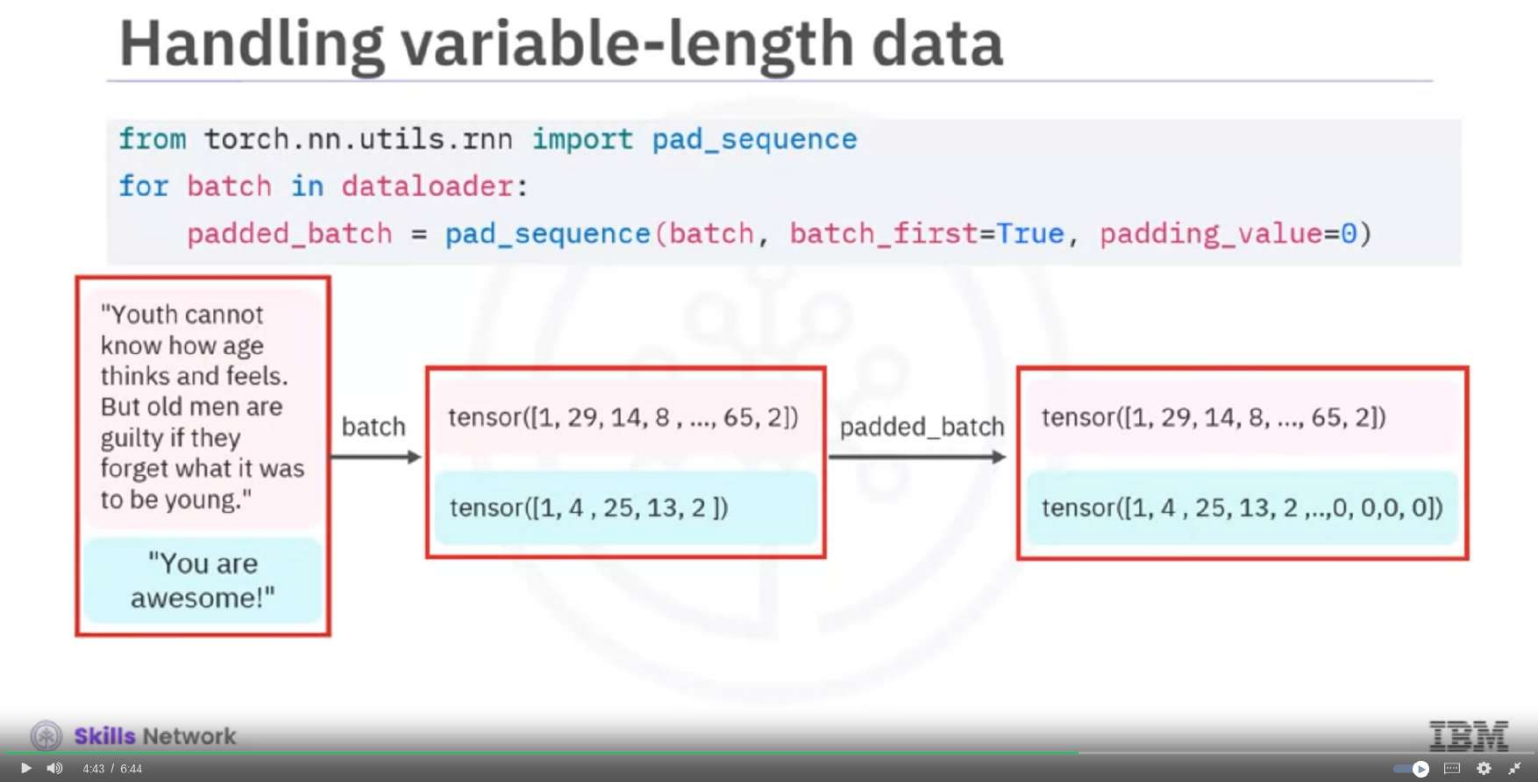
Tokenization and vocabulary building

```python
tokenizer = get_tokenizer("basic_english")
vocab = build_vocab_from_iterator(map(tokenizer, sentences))
```

"You are awesome!"

Tokenizer

You	are	awesome	!
19	4	25	20



Handling variable-length data

```
from torch.nn.utils.rnn import pad_sequence
for batch in dataloader:
    padded_batch = pad_sequence(batch, batch_first=True, padding_value=0)
```

"Youth cannot know how age thinks and feels. But old men are guilty if they forget what it was to be young."

"You are awesome!"

batch

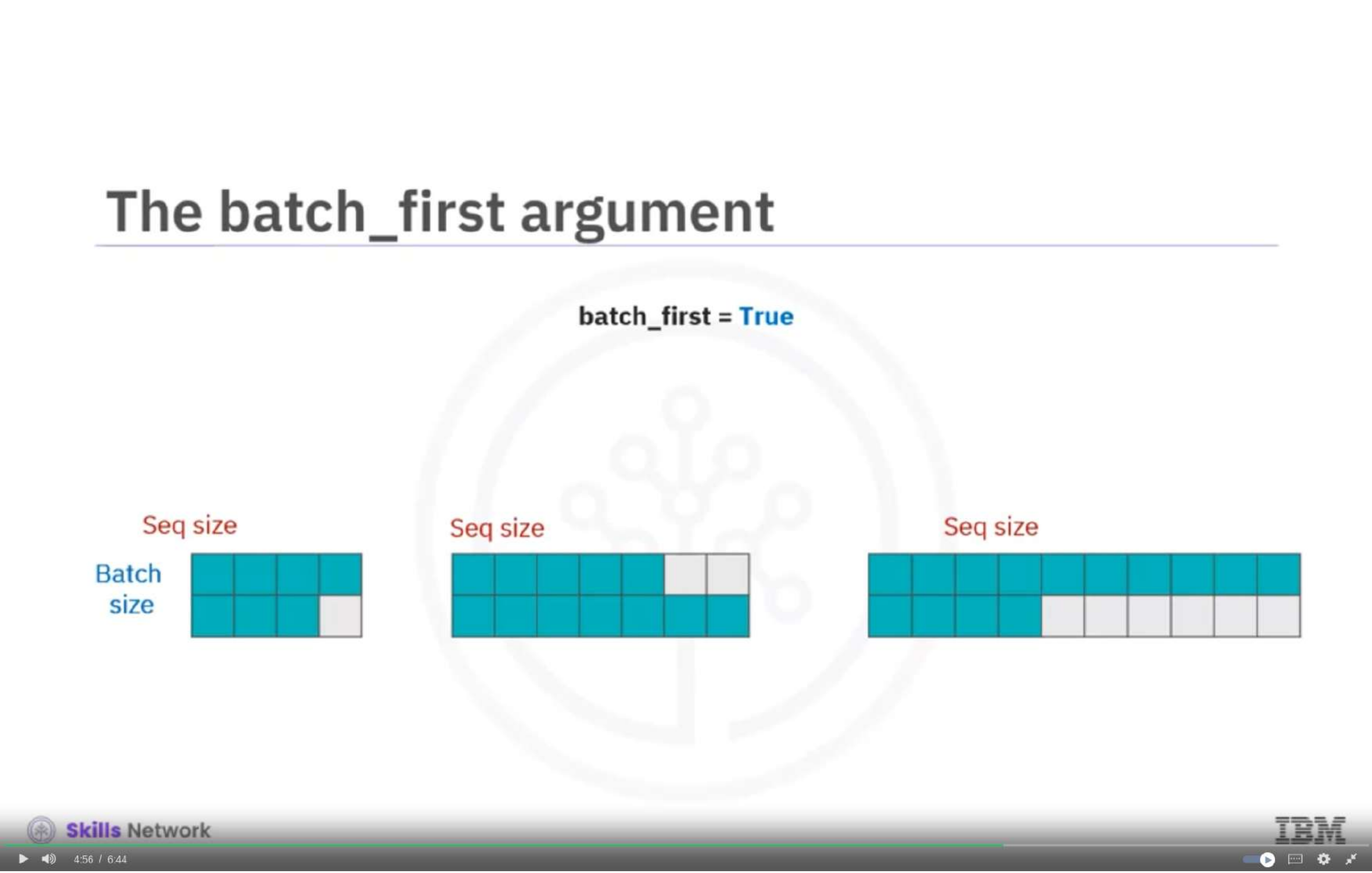
tensor([1, 29, 14, 8 , ..., 65, 2])

tensor([1, 4 , 25, 13, 2])

padded_batch

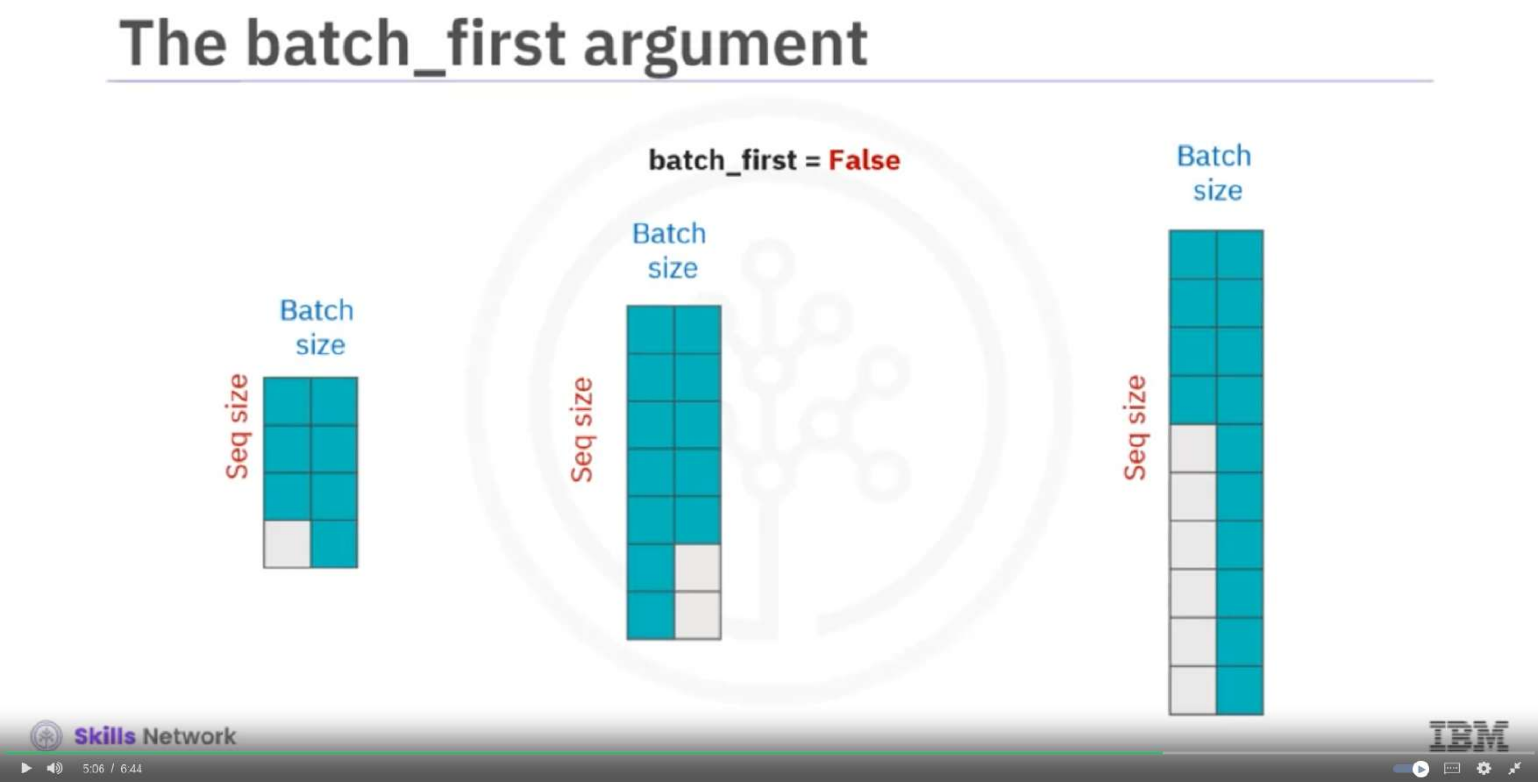
tensor([1, 29, 14, 8, ..., 65, 2])

tensor([1, 4 , 25, 13, 2 ,..,0, 0,0, 0])



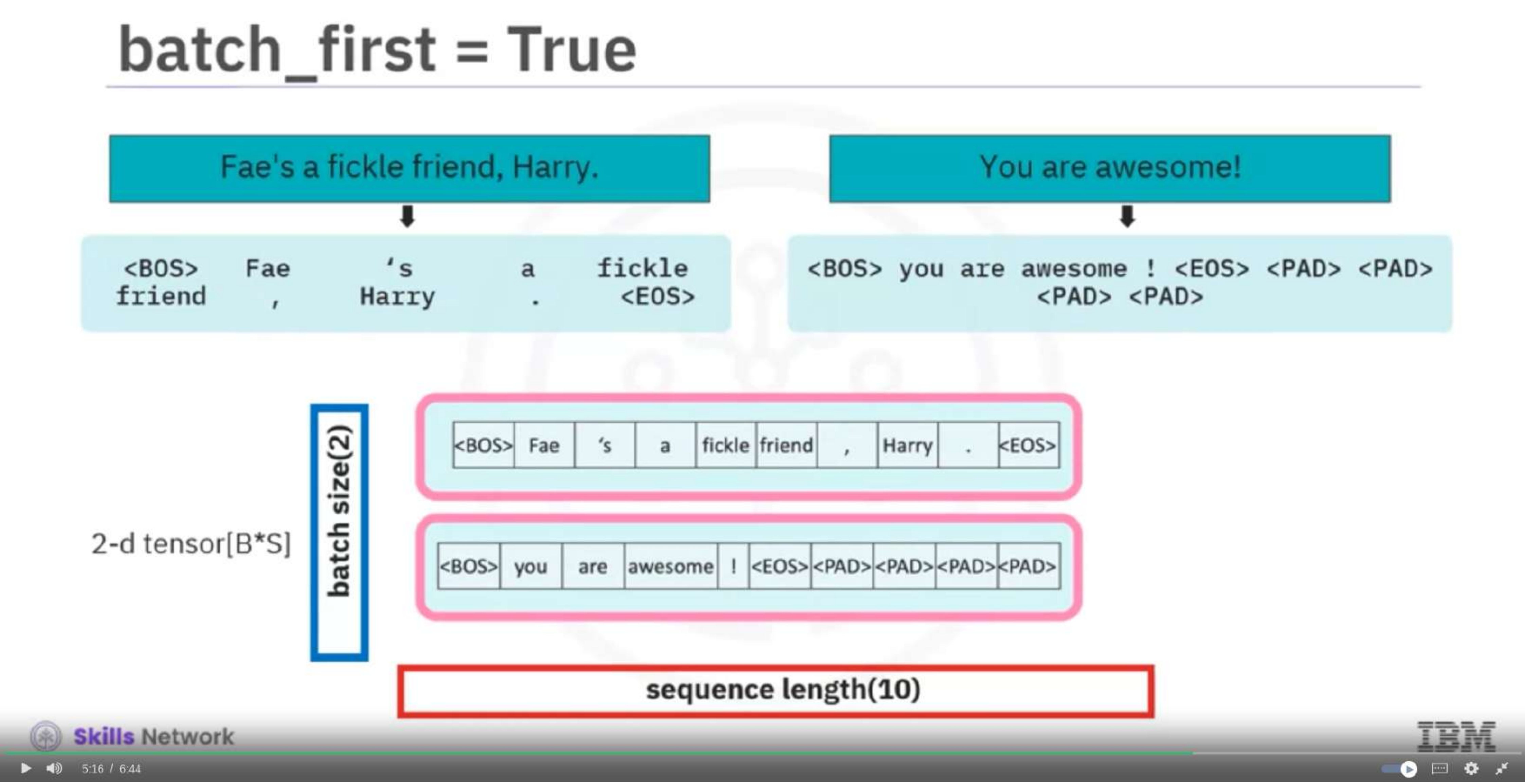
The batch_first argument

batch_first = True



The batch_first argument

batch_first = False



batch_first = True

Fae's a fickle friend, Harry.

<BOS> Fae ‘s a fickle friend , Harry . <EOS>

You are awesome!

<BOS> you are awesome ! <EOS> <PAD> <PAD> <PAD> <PAD>

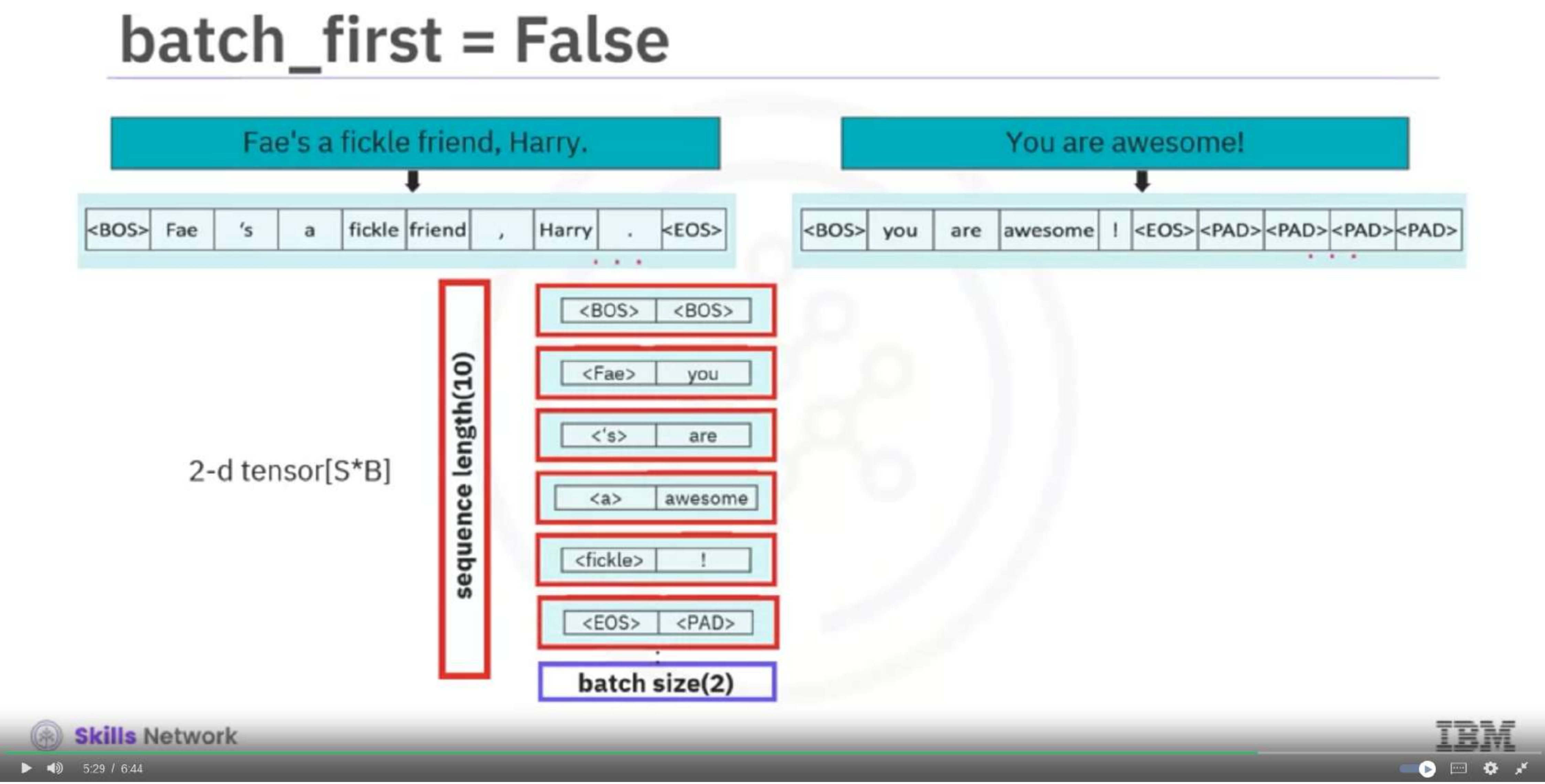
2-d tensor[B*S]

batch size(2)

<BOS>	Fae	‘s	a	fickle	friend	,	Harry	.	<EOS>

<BOS>	you	are	awesome	!	<EOS>	<PAD>	<PAD>	<PAD>	<PAD>

sequence length(10)



batch_first = False

Fae's a fickle friend, Harry.

<BOS>	Fae	's	a	fickle	friend	,	Harry	.	<EOS>

You are awesome!

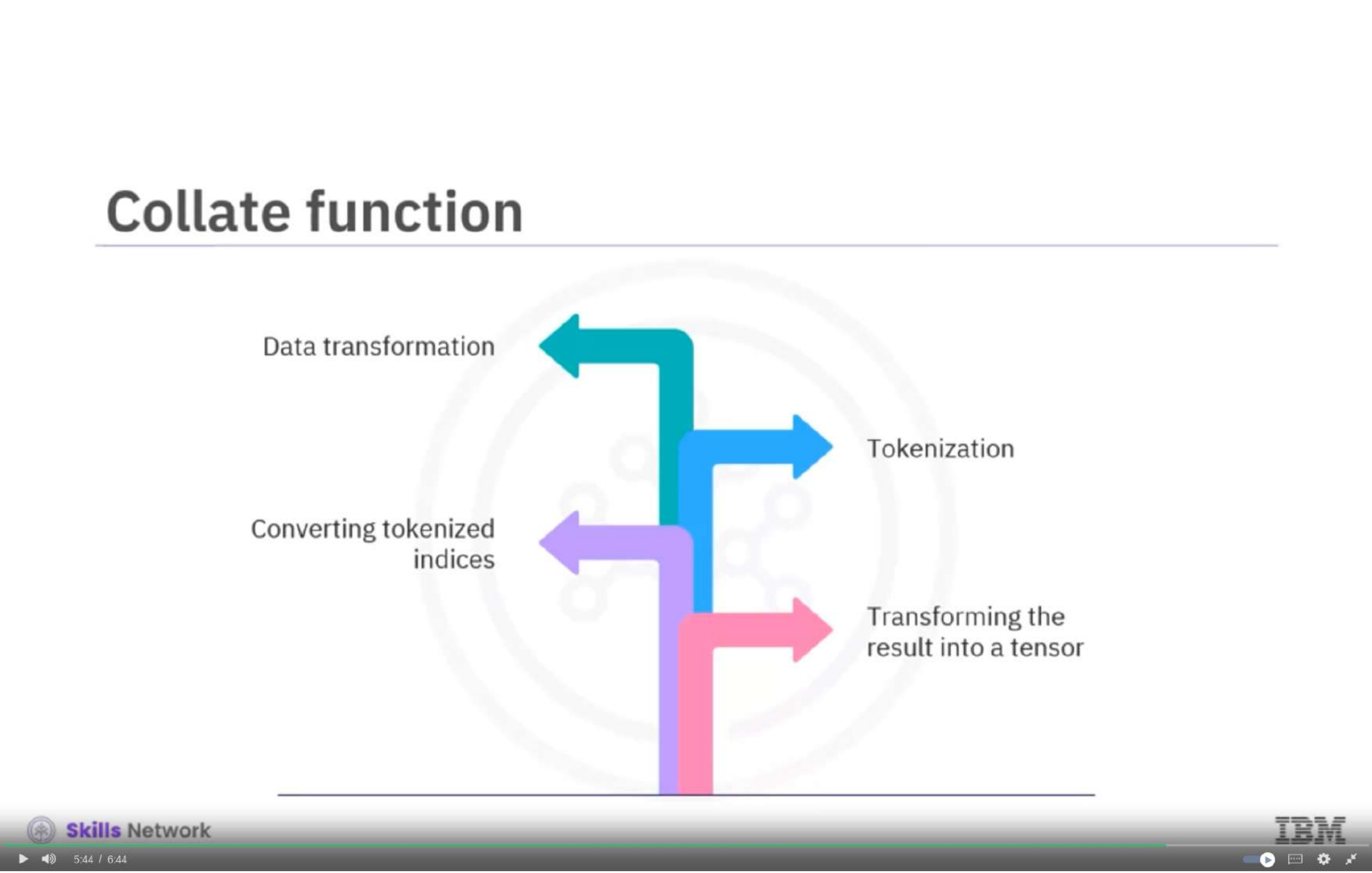
<BOS>	you	are	awesome	!	<EOS>	<PAD>	<PAD>	<PAD>	<PAD>

2-d tensor[S*B]

sequence length(10)

<BOS>	<BOS>
<Fae>	you
<'s>	are
<a>	awesome
<fickle>	!
<EOS>	<PAD>

batch size(2)



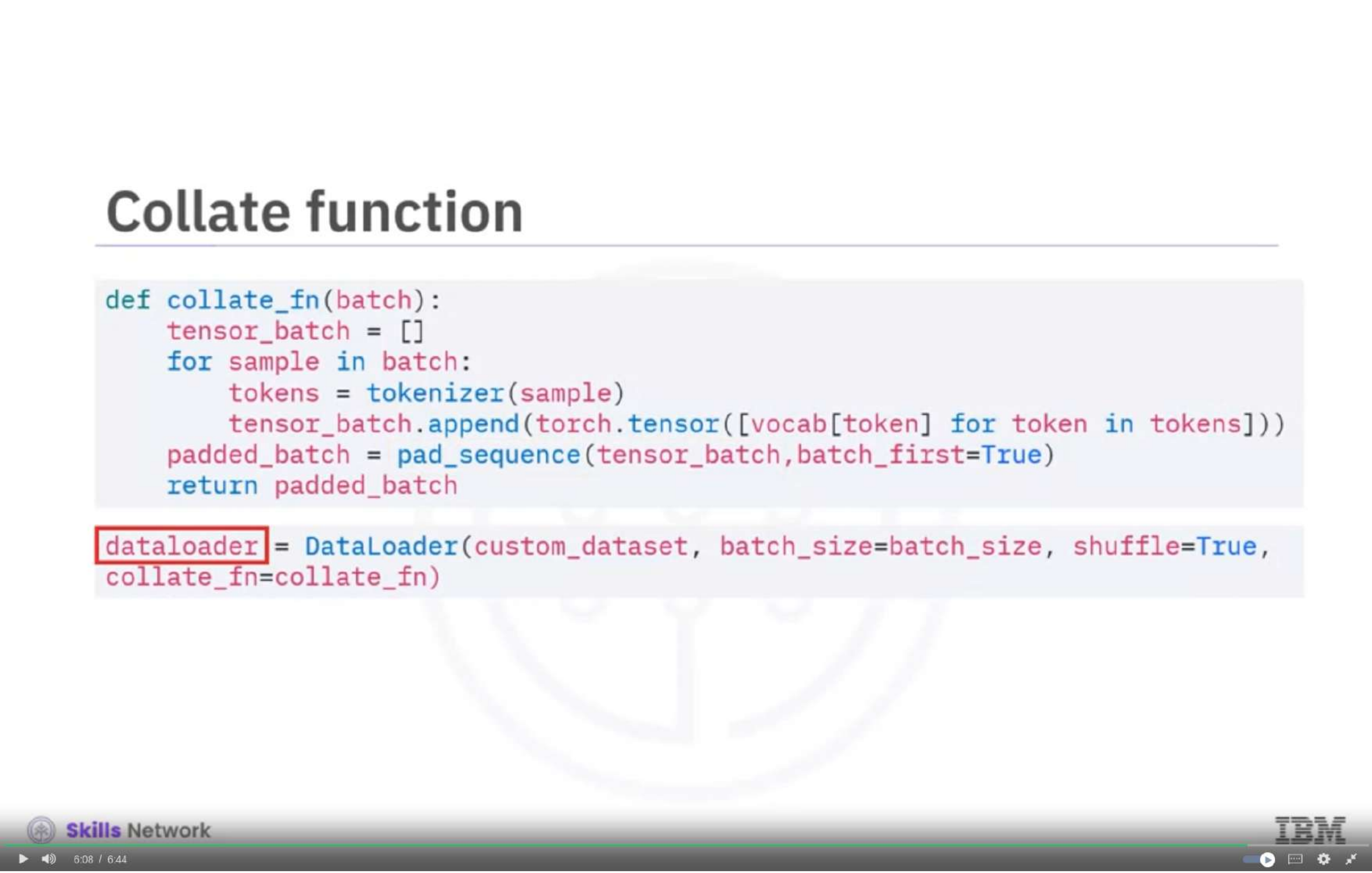
Collate function

Data transformation

Tokenization

Converting tokenized indices

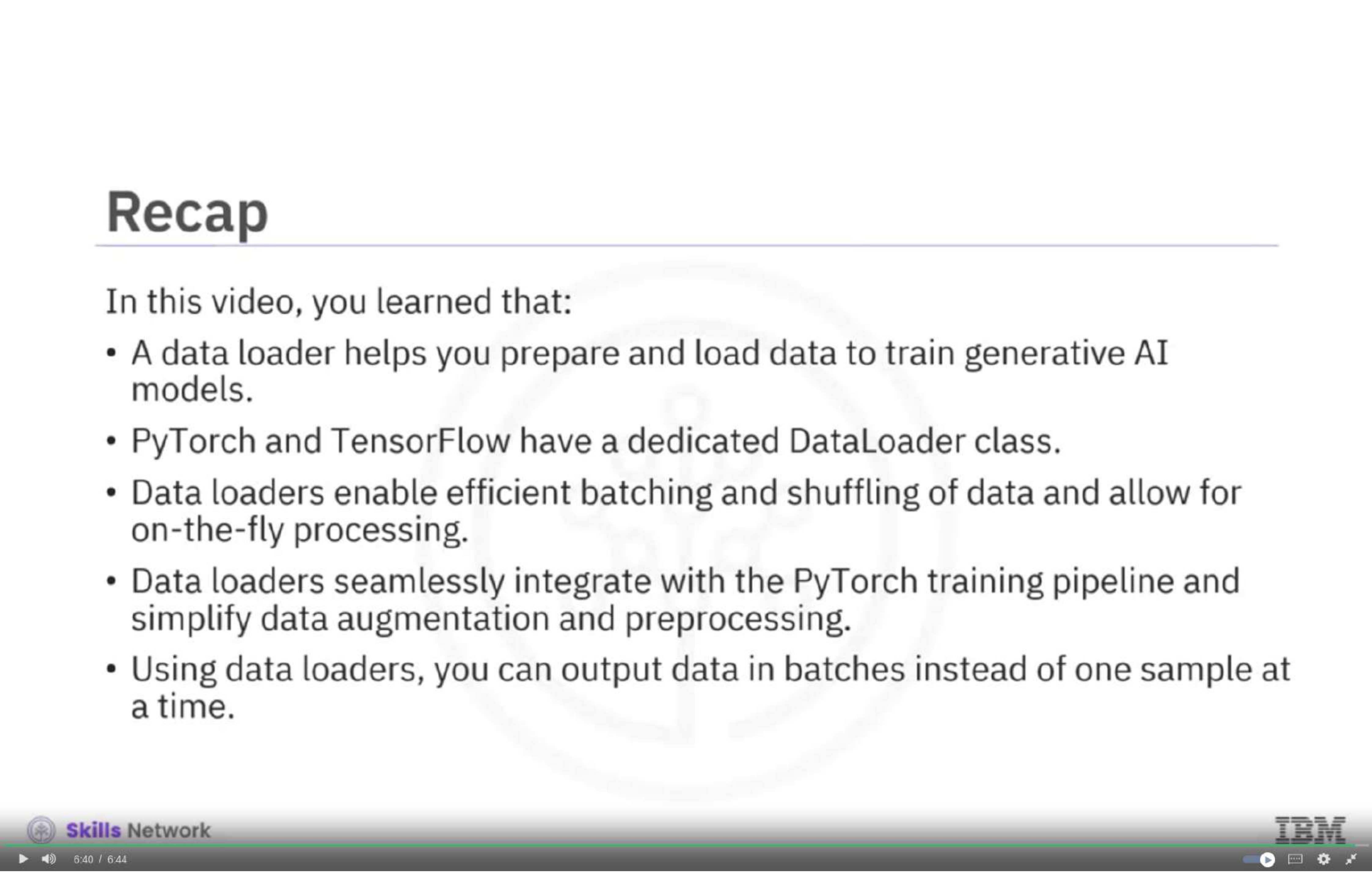
Transforming the result into a tensor



Collate function

```
def collate_fn(batch):
    tensor_batch = []
    for sample in batch:
        tokens = tokenizer(sample)
        tensor_batch.append(torch.tensor([vocab[token] for token in tokens]))
    padded_batch = pad_sequence(tensor_batch,batch_first=True)
    return padded_batch
```

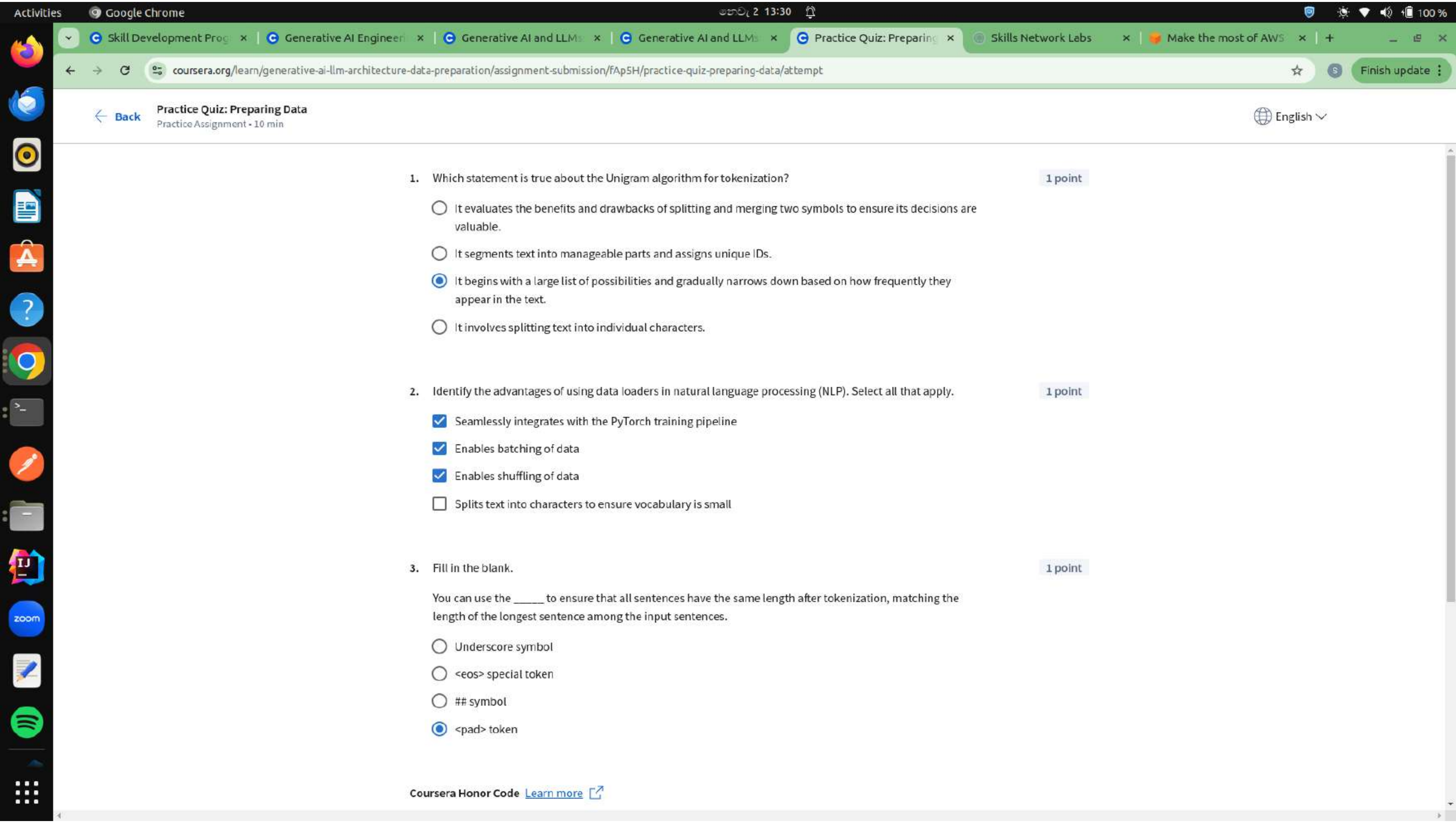
```
dataloader = DataLoader(custom_dataset, batch_size=batch_size, shuffle=True,
collate_fn=collate_fn)
```



Recap

In this video, you learned that:

- A data loader helps you prepare and load data to train generative AI models.
- PyTorch and TensorFlow have a dedicated DataLoader class.
- Data loaders enable efficient batching and shuffling of data and allow for on-the-fly processing.
- Data loaders seamlessly integrate with the PyTorch training pipeline and simplify data augmentation and preprocessing.
- Using data loaders, you can output data in batches instead of one sample at a time.



Practice Quiz: Preparing Data

Practice Assignment • 10 min

1. Which statement is true about the Unigram algorithm for tokenization?

1 point

○ It evaluates the benefits and drawbacks of splitting and merging two symbols to ensure its decisions are valuable.

○ It segments text into manageable parts and assigns unique IDs.

● It begins with a large list of possibilities and gradually narrows down based on how frequently they appear in the text.

○ It involves splitting text into individual characters.

2. Identify the advantages of using data loaders in natural language processing (NLP). Select all that apply.

1 point

☑ Seamlessly integrates with the PyTorch training pipeline

☑ Enables batching of data

☑ Enables shuffling of data

□ Splits text into characters to ensure vocabulary is small

3. Fill in the blank.

1 point

You can use the _____ to ensure that all sentences have the same length after tokenization, matching the length of the longest sentence among the input sentences.

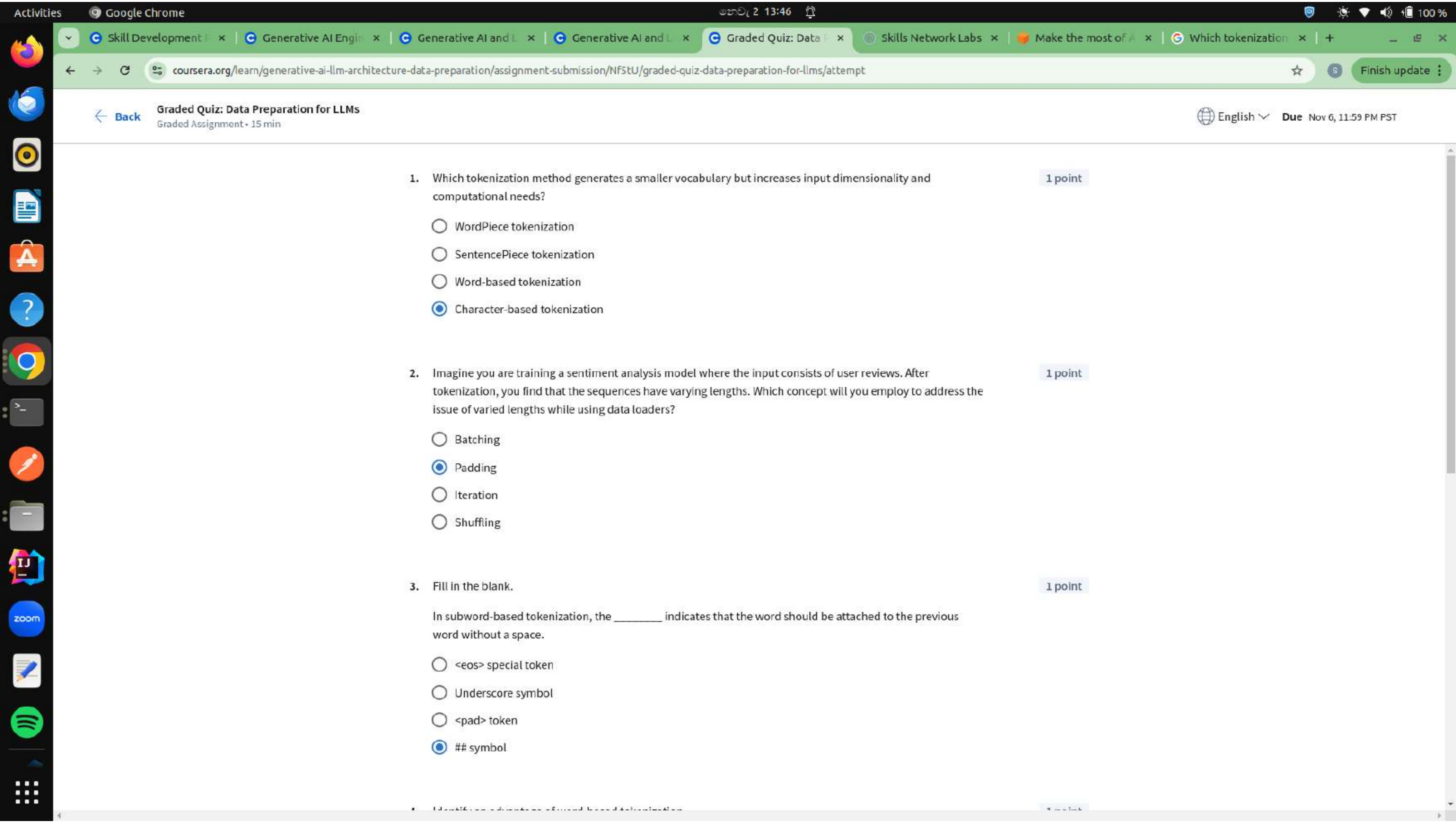
○ Underscore symbol

○ <eos> special token

○ ## symbol

● <pad> token

Coursera Honor Code Learn more



Graded Quiz: Data Preparation for LLMs

Graded Assignment • 15 min

Due Nov 6, 11:59 PM PST

1. Which tokenization method generates a smaller vocabulary but increases input dimensionality and computational needs? 1 point

○ WordPiece tokenization

○ SentencePiece tokenization

○ Word-based tokenization

● Character-based tokenization

2. Imagine you are training a sentiment analysis model where the input consists of user reviews. After tokenization, you find that the sequences have varying lengths. Which concept will you employ to address the issue of varied lengths while using data loaders? 1 point

○ Batching

● Padding

○ Iteration

○ Shuffling

3. Fill in the blank. 1 point

In subword-based tokenization, the ________ indicates that the word should be attached to the previous word without a space.

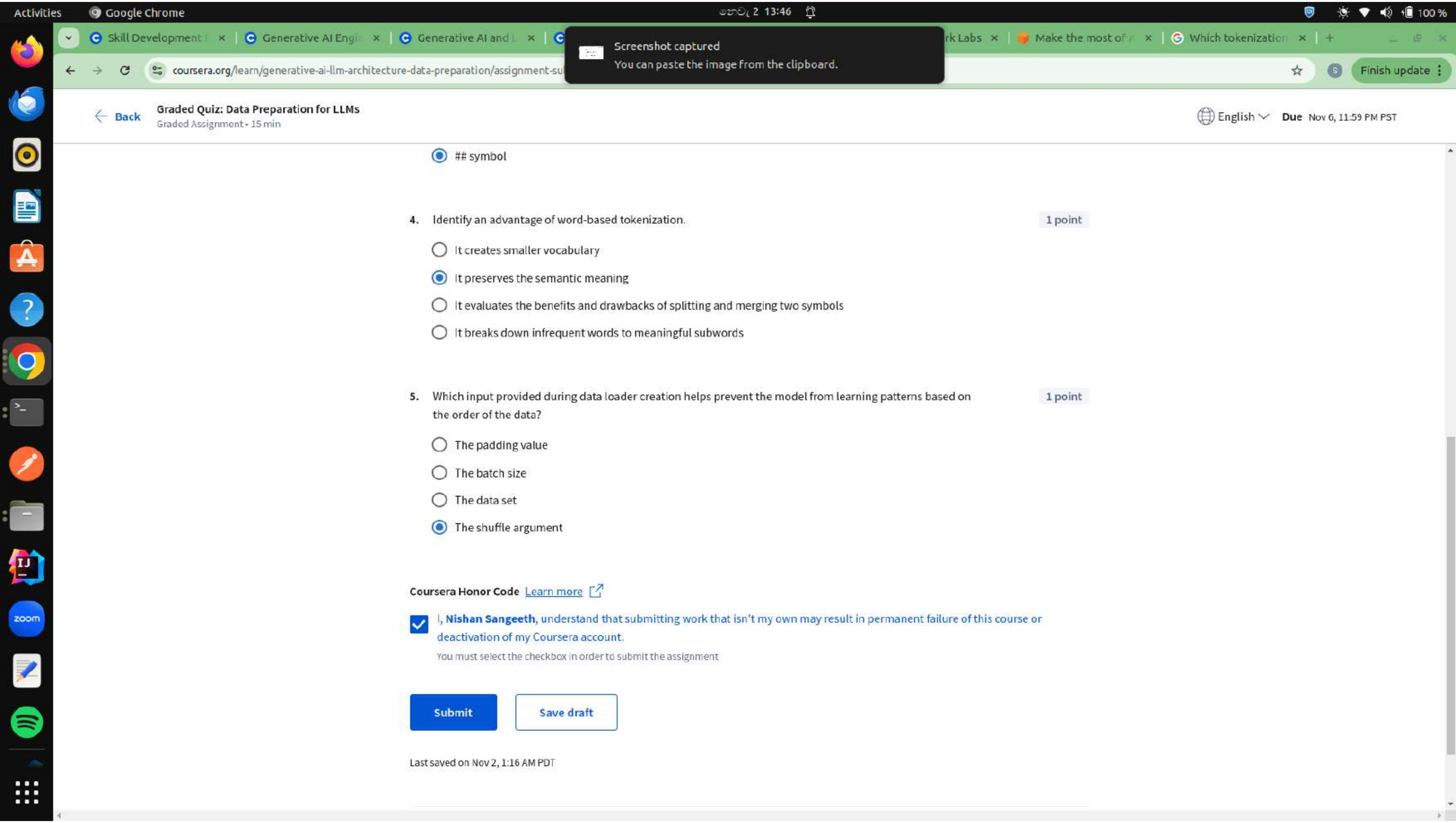
○ <eos> special token

○ Underscore symbol

○ <pad> token

● ## symbol

4. Identify an advantage of word-based tokenization 1 point



Graded Quiz: Data Preparation for LLMs

Graded Assignment • 15 min

● ## symbol

4. Identify an advantage of word-based tokenization. 1 point

○ It creates smaller vocabulary

● It preserves the semantic meaning

○ It evaluates the benefits and drawbacks of splitting and merging two symbols

○ It breaks down infrequent words to meaningful subwords

5. Which input provided during data loader creation helps prevent the model from learning patterns based on the order of the data? 1 point

○ The padding value

○ The batch size

○ The data set

● The shuffle argument

Coursera Honor Code Learn more

☑ I, **Nishan Sangeeth**, understand that submitting work that isn't my own may result in permanent failure of this course or deactivation of my Coursera account.

You must select the checkbox in order to submit the assignment

Submit Save draft

Last saved on Nov 2, 1:16 AM PDT