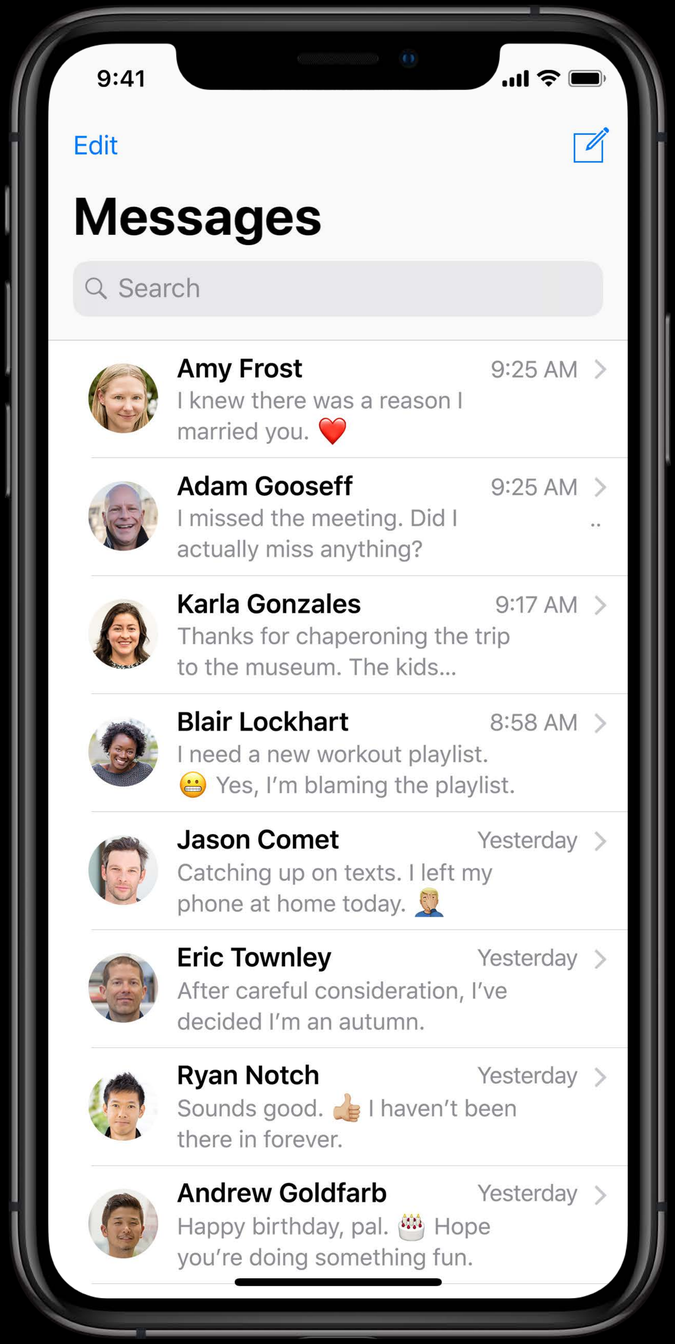
#WWDC19

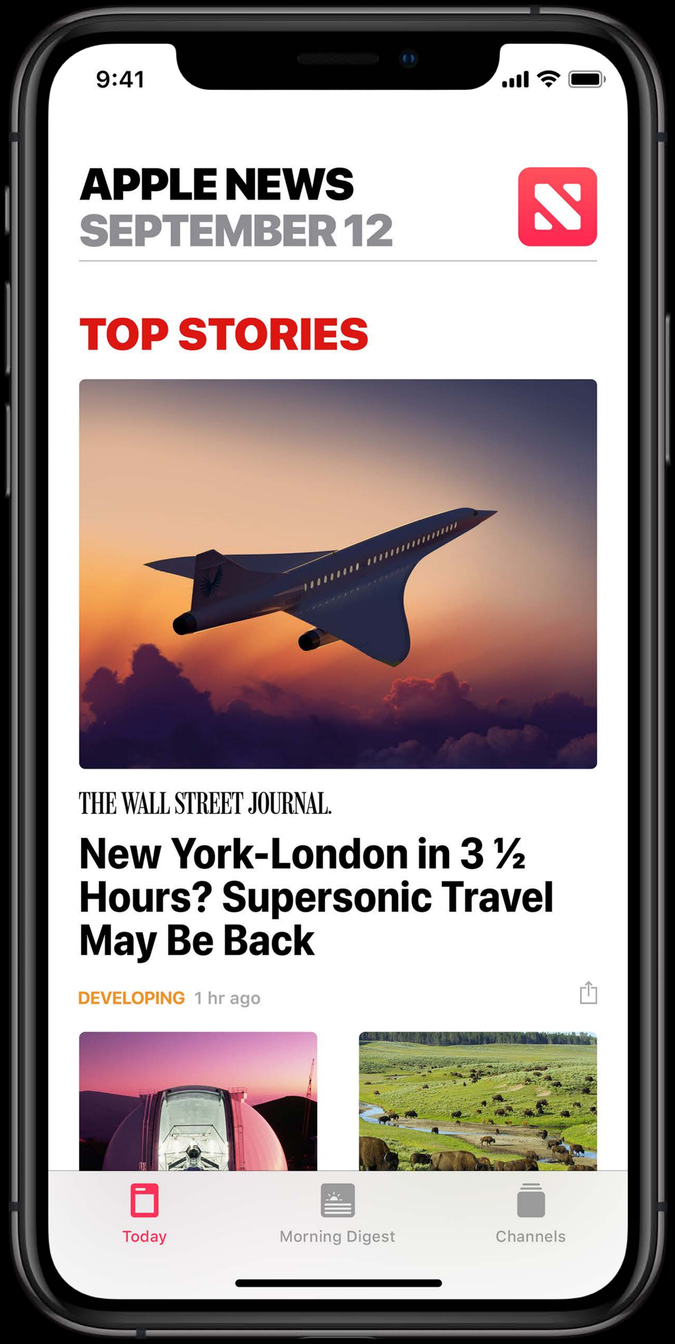
Advances in Natural Language Framework

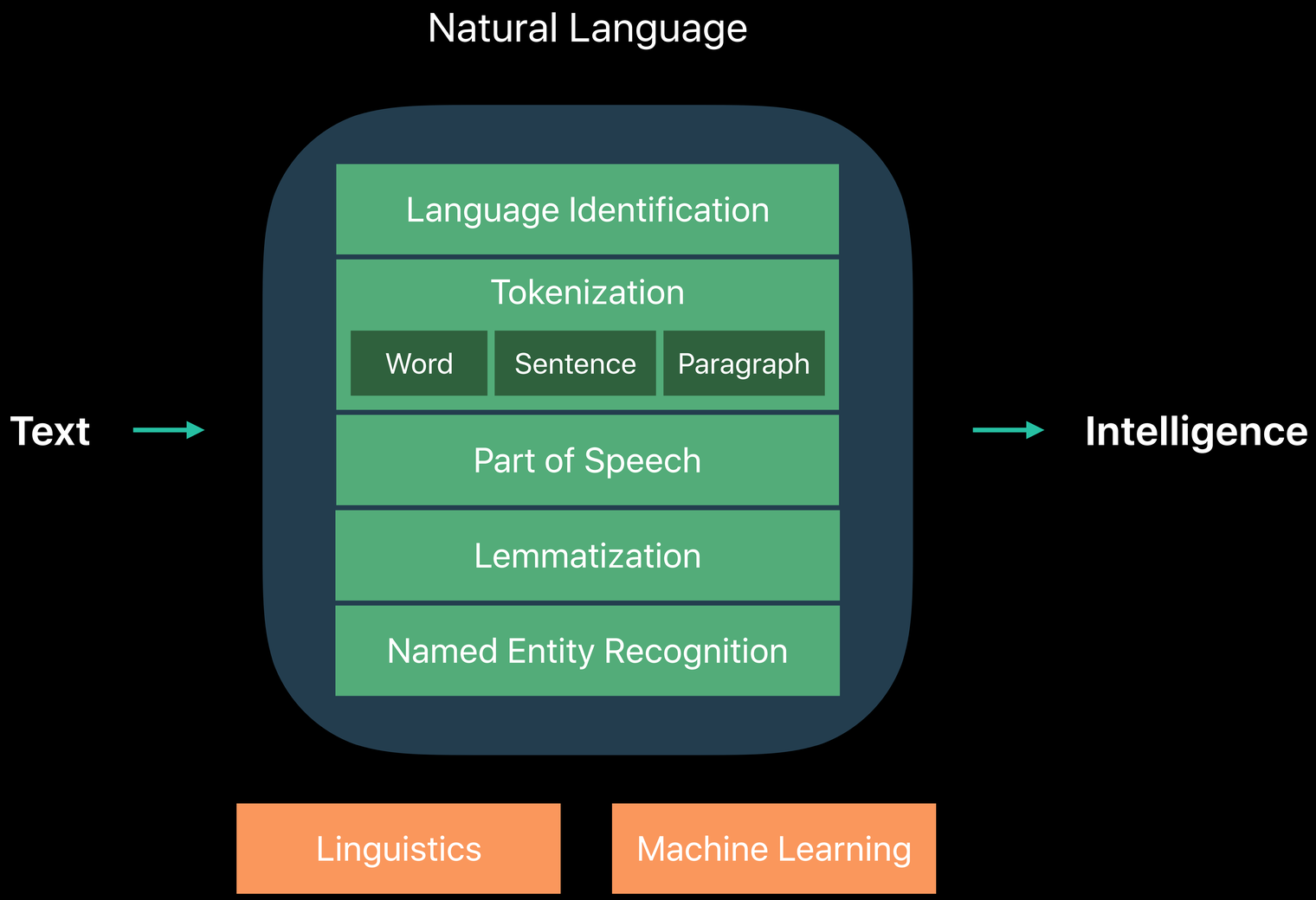
Vivek Kumar Rangarajan Sridhar, NLP Technologies
Doug Davidson, NLP Technologies

Natural
language
input


9:41
Edit
Messages
Search
Amy Frost
9:25 AM
I knew there was a reason I married you. ❤️
Adam Gooseff
9:25 AM
I missed the meeting. Did I actually miss anything?
Karla Gonzales
9:17 AM
Thanks for chaperoning the trip to the museum. The kids...
Blair Lockhart
8:58 AM
I need a new workout playlist. 😬 Yes, I'm blaming the playlist.
Jason Comet
Yesterday
Catching up on texts. I left my phone at home today. 🤦
Eric Townley
Yesterday
After careful consideration, I've decided I'm an autumn.
Ryan Notch
Yesterday
Sounds good. 👍 I haven't been there in forever.
Andrew Goldfarb
Yesterday
Happy birthday, pal. 🎂 Hope you're doing something fun.

- Natural language output


9:41
APPLE NEWS
SEPTEMBER 12
TOP STORIES
THE WALL STREET JOURNAL.
New York-London in 3 ½ Hours? Supersonic Travel May Be Back
DEVELOPING 1 hr ago
Today
Morning Digest
Channels


Natural Language
Language Identification
Tokenization
Word
Sentence
Paragraph
Part of Speech
Lemmatization
Named Entity Recognition
Text
Intelligence
Linguistics
Machine Learning

Text Classification

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Label

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Label

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Label

Text Classification

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Label

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Label

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

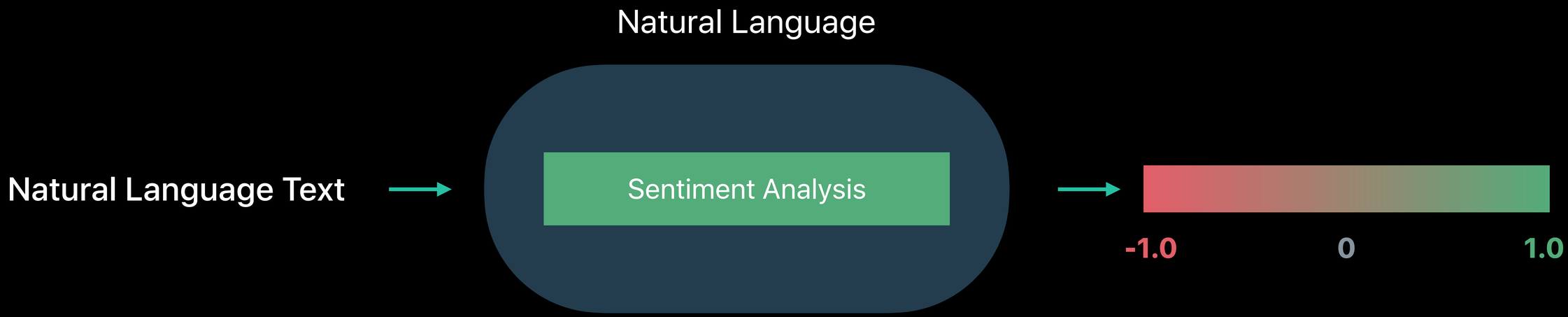
Label

Word Tagging

Lorem	ipsum	dolor	sit	amet
Label	Label	Label	Label	Label

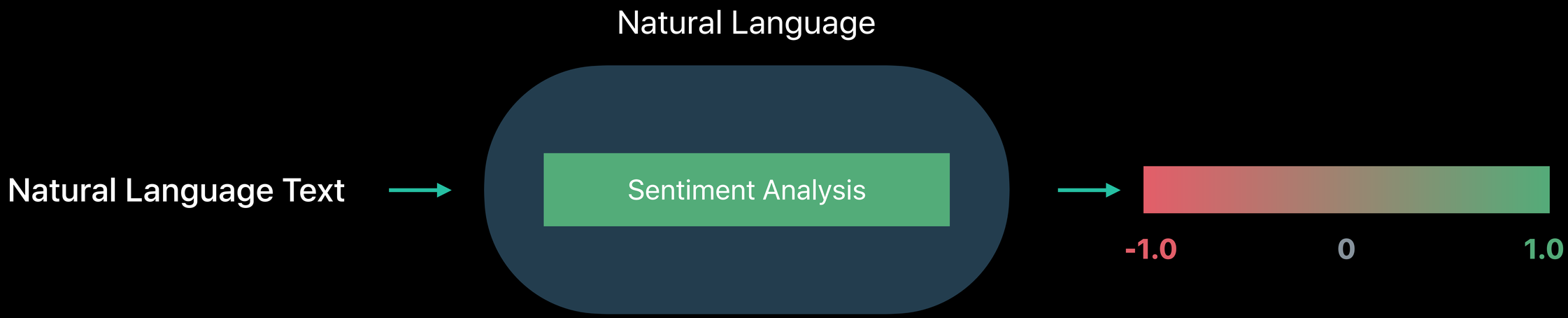
Sentiment Analysis

Text Classification



Sentiment Analysis

Text Classification

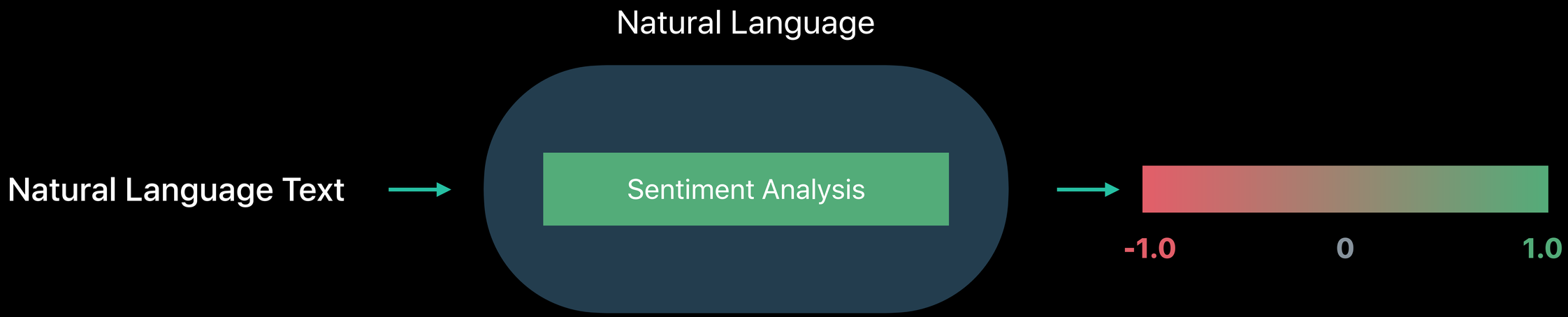


“We had a fun time in Hawaii with the family!”

0.8

Sentiment Analysis

Text Classification



“We had a not so fun time in Hawaii cause mom twisted her ankle.”

-0.8

Sentiment Analysis

```
import NaturalLanguage

let tagger = NLTagger(tagSchemes: [.sentimentScore])

tagger.string = string

let (sentiment, _) = tagger.tag(at: string.startIndex, unit: .paragraph, scheme: .sentimentScore)
```

Sentiment Analysis

```
import NaturalLanguage

let tagger = NLTagger(tagSchemes: [.sentimentScore])

tagger.string = string

let (sentiment, _) = tagger.tag(at: string.startIndex, unit: .paragraph, scheme: .sentimentScore)
```

Sentiment Analysis

```
import NaturalLanguage

let tagger = NLTagger(tagSchemes: [.sentimentScore])

tagger.string = string

let (sentiment, _) = tagger.tag(at: string.startIndex, unit: .paragraph, scheme: .sentimentScore)
```

Sentiment Analysis

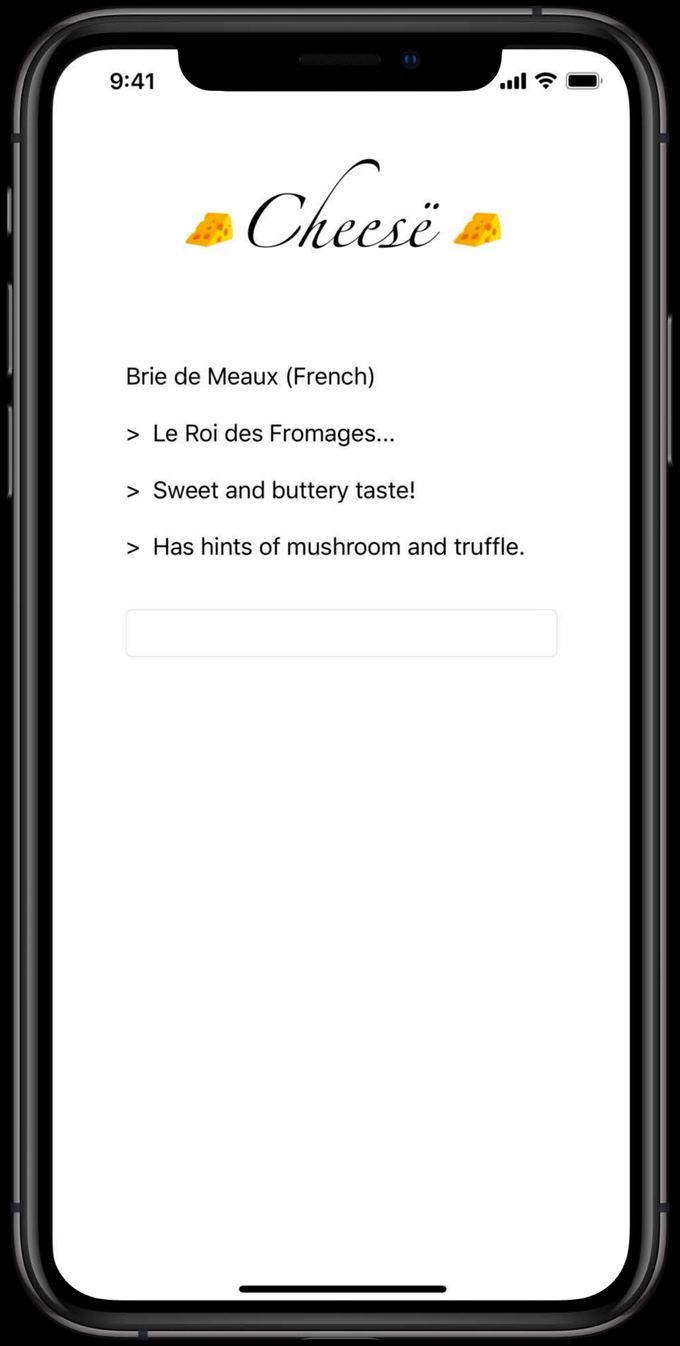
```
import NaturalLanguage

let tagger = NLTagger(tagSchemes: [.sentimentScore])

tagger.string = string

let (sentiment, _) = tagger.tag(at: string.startIndex, unit: .paragraph, scheme: .sentimentScore)
```

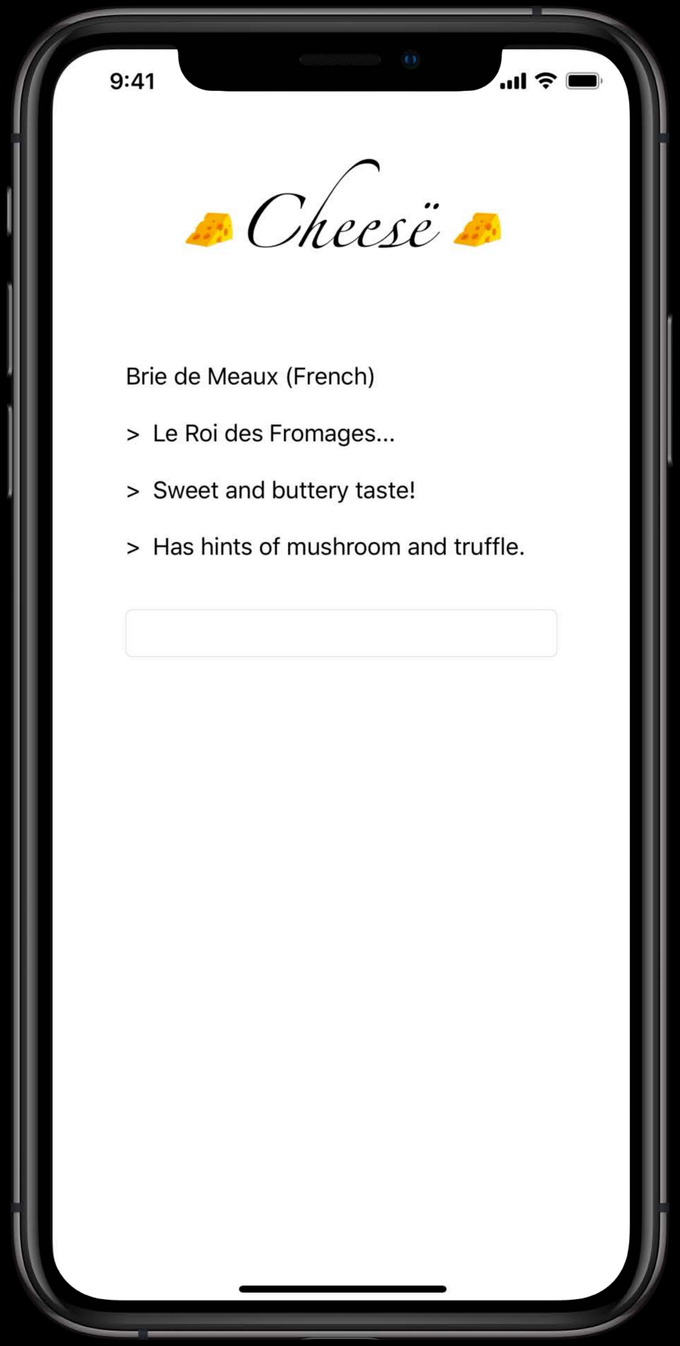
Sentiment Analysis

```
import NaturalLanguage

let tagger = NLTagger(tagSchemes: [.sentimentScore])

tagger.string = string

let (sentiment, _) = tagger.tag(at: string.startIndex, unit: .paragraph, scheme: .sentimentScore)
```

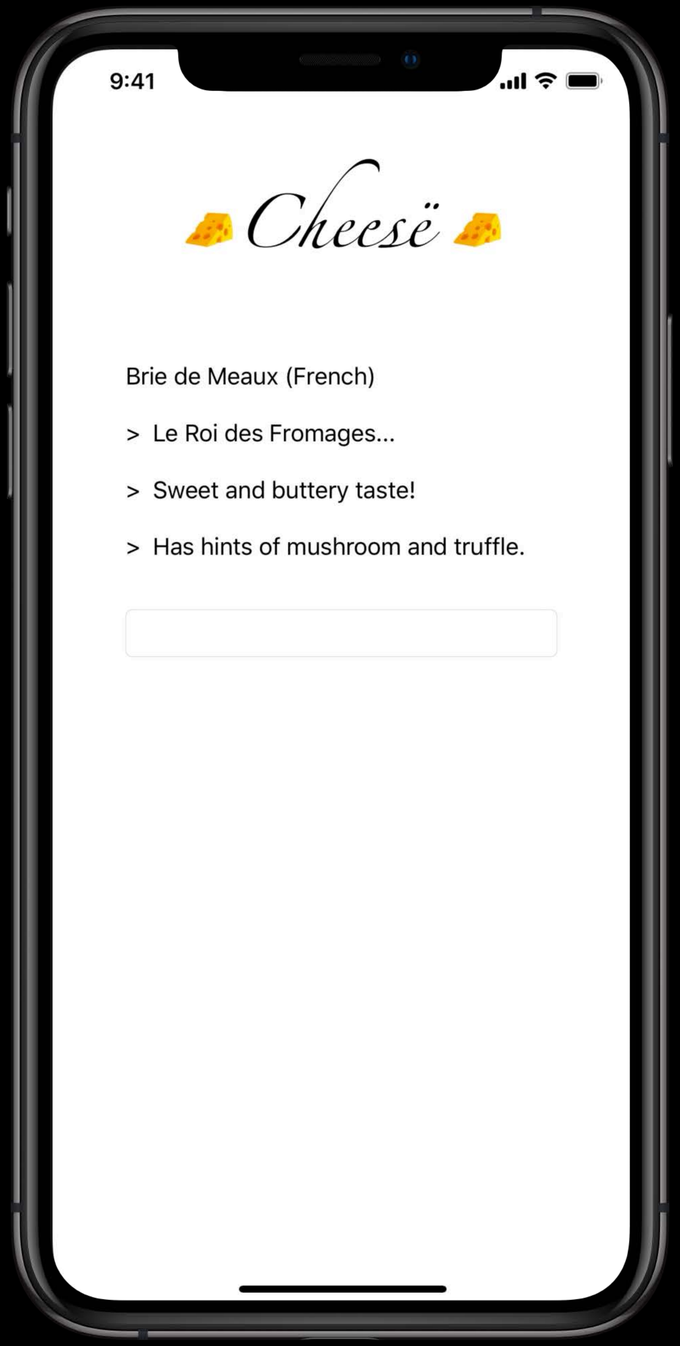
Sentiment Analysis

```
import NaturalLanguage

let tagger = NLTagger(tagSchemes: [.sentimentScore])

tagger.string = string

let (sentiment, _) = tagger.tag(at: string.startIndex, unit: .paragraph, scheme: .sentimentScore)
```


9:41
Cheesë
Brie de Meaux (French)
> Le Roi des Fromages...
> Sweet and buttery taste!
> Has hints of mushroom and truffle.


9:41
Cheesë
Brie de Meaux (French)
> Le Roi des Fromages...
> Sweet and buttery taste!
> Has hints of mushroom and truffle.

Performance



7 Languages

Language Assets

Language Assets

```
import NaturalLanguage

NLTagger.requestAssets(for: .french, tagScheme: .sentimentScore) { (result, error) in
  // handle result
}
```

Text Classification

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Label

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Label

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Label

Word Tagging

Lorem	ipsum	dolor	sit	amet
Label	Label	Label	Label	Label

Word Tagging

Timothy and I visited Switzerland this summer and fell in love with Gruyère cheese

Word Tagging

Timothy and I visited Switzerland this summer and fell in love with Gruyère cheese

Person	Location	Noun	Noun	Noun	Noun

Word Tagging

Timothy and I visited Switzerland this summer and fell in love with Gruyère cheese

Person	Location	Noun	Noun	Swiss cheese

Text Catalog

Word Tagging

Text Catalog

Creation

```
import CreateML

let entities = ["Italian Cheese": ["Parmigiano-Reggiano", "Pecorino Romano", "Montasio", …],
                "Mexican Cheese": ["Cotija", "Añejo", "Chihuahua", "Queso de cuajo", …],
                "American Cheese": ["Monterey Jack", "Colby cheese", "Colorado Blackie", …],
                "Greek Cheese": ["Feta", "Kasseri", "Manouri", "Kefalotyri", …], …]

let gazetteer = try MLGazetteer(dictionary: entities)

try gazetteer.write(to: url)
```

Text Catalog

Creation

```
import CreateML

let entities = ["Italian Cheese": ["Parmigiano-Reggiano", "Pecorino Romano", "Montasio", …],
                "Mexican Cheese": ["Cotija", "Añejo", "Chihuahua", "Queso de cuajo", …],
                "American Cheese": ["Monterey Jack", "Colby cheese", "Colorado Blackie", …],
                "Greek Cheese": ["Feta", "Kasseri", "Manouri", "Kefalotyri", …], …]

let gazetteer = try MLGazetteer(dictionary: entities)

try gazetteer.write(to: url)
```

Text Catalog

Creation

```
import CreateML

let entities = ["Italian Cheese": ["Parmigiano-Reggiano", "Pecorino Romano", "Montasio", …],
                "Mexican Cheese": ["Cotija", "Añejo", "Chihuahua", "Queso de cuajo", …],
                "American Cheese": ["Monterey Jack", "Colby cheese", "Colorado Blackie", …],
                "Greek Cheese": ["Feta", "Kasseri", "Manouri", "Kefalotyri", …], …]

let gazetteer = try MLGazetteer(dictionary: entities)

try gazetteer.write(to: url)
```

Text Catalog

Creation

```
import CreateML

let entities = ["Italian Cheese": ["Parmigiano-Reggiano", "Pecorino Romano", "Montasio", …],
                "Mexican Cheese": ["Cotija", "Añejo", "Chihuahua", "Queso de cuajo", …],
                "American Cheese": ["Monterey Jack", "Colby cheese", "Colorado Blackie", …],
                "Greek Cheese": ["Feta", "Kasseri", "Manouri", "Kefalotyri", …], …]

let gazetteer = try MLGazetteer(dictionary: entities)

try gazetteer.write(to: url)
```

Text Catalog

Creation

```
import CreateML

let entities = ["Italian Cheese": ["Parmigiano-Reggiano", "Pecorino Romano", "Montasio", …],
                "Mexican Cheese": ["Cotija", "Añejo", "Chihuahua", "Queso de cuajo", …],
                "American Cheese": ["Monterey Jack", "Colby cheese", "Colorado Blackie", …],
                "Greek Cheese": ["Feta", "Kasseri", "Manouri", "Kefalotyri", …], …]

let gazetteer = try MLGazetteer(dictionary: entities)

try gazetteer.write(to: url)
```

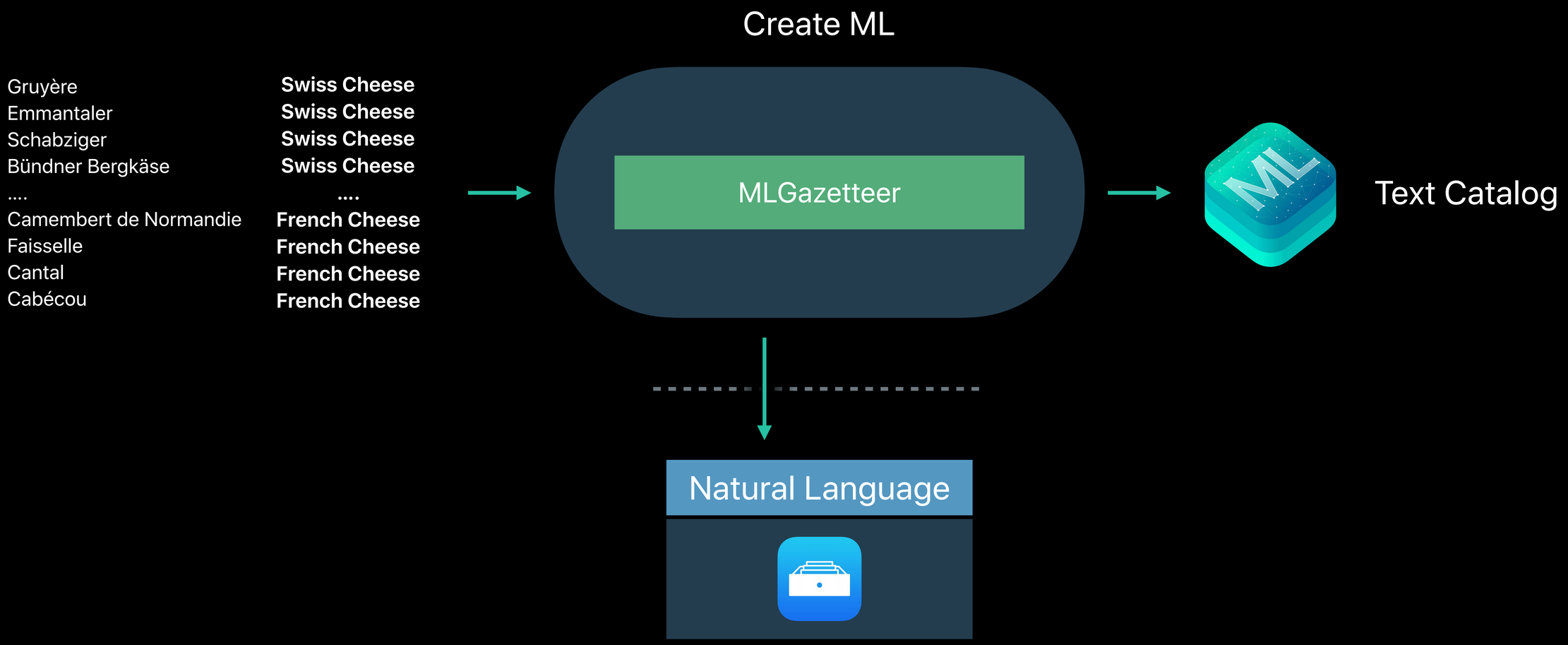
Text Catalog

Creation

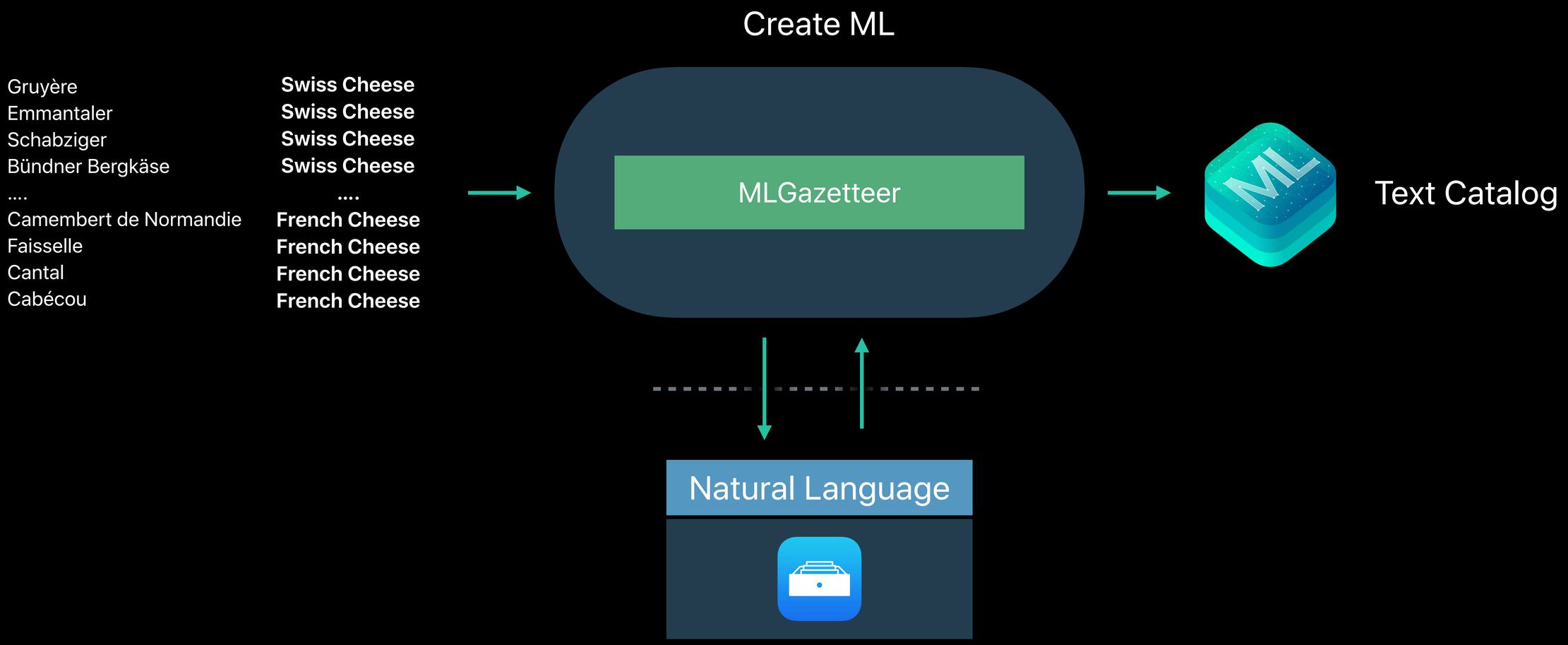
```
import CreateML

let entities = ["Italian Cheese": ["Parmigiano-Reggiano", "Pecorino Romano", "Montasio", …],
                "Mexican Cheese": ["Cotija", "Añejo", "Chihuahua", "Queso de cuajo", …],
                "American Cheese": ["Monterey Jack", "Colby cheese", "Colorado Blackie", …],
                "Greek Cheese": ["Feta", "Kasseri", "Manouri", "Kefalotyri", …], …]

let gazetteer = try MLGazetteer(dictionary: entities)

try gazetteer.write(to: url)
```

Under the Hood

Under the Hood



Under the Hood



Text Catalog

Usage with a tagger

```
import NaturalLanguage

let gazetteer = try! NLGazetteer(contentsOf: url)

let tagger = NLTagger(tagSchemes: [.nameTypeOrLexicalClass])

tagger.setGazetteers([gazetteer], for: .nameTypeOrLexicalClass)
```

Text Catalog

Usage with a tagger

```
import NaturalLanguage

let gazetteer = try! NLGazetteer(contentsOf: url)

let tagger = NLTagger(tagSchemes: [.nameTypeOrLexicalClass])

tagger.setGazetteers([gazetteer], for: .nameTypeOrLexicalClass)
```

Text Catalog

Usage with a tagger

```
import NaturalLanguage

let gazetteer = try! NLGazetteer(contentsOf: url)

let tagger = NLTagger(tagSchemes: [.nameTypeOrLexicalClass])

tagger.setGazetteers([gazetteer], for: .nameTypeOrLexicalClass)
```

Text Catalog

Usage with a tagger

```
import NaturalLanguage

let gazetteer = try! NLGazetteer(contentsOf: url)

let tagger = NLTagger(tagSchemes: [.nameTypeOrLexicalClass])

tagger.setGazetteers([gazetteer], for: .nameTypeOrLexicalClass)
```

Text Catalog

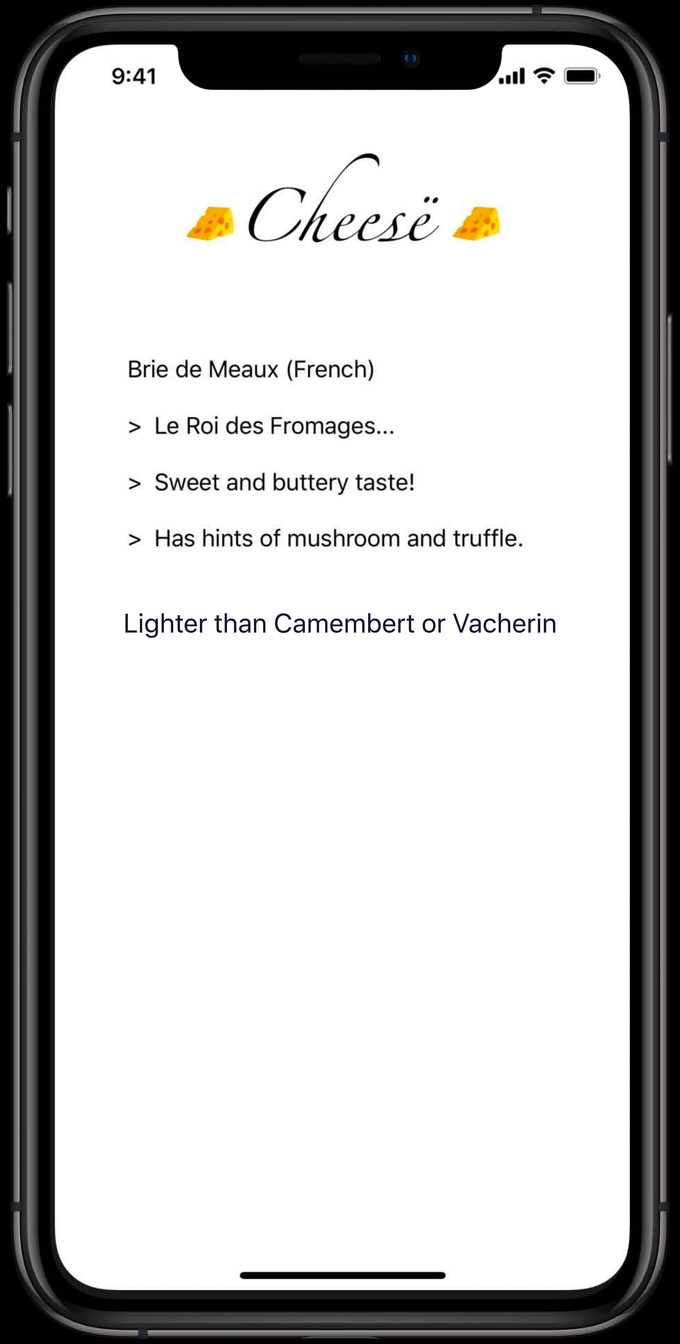
Usage with a tagger

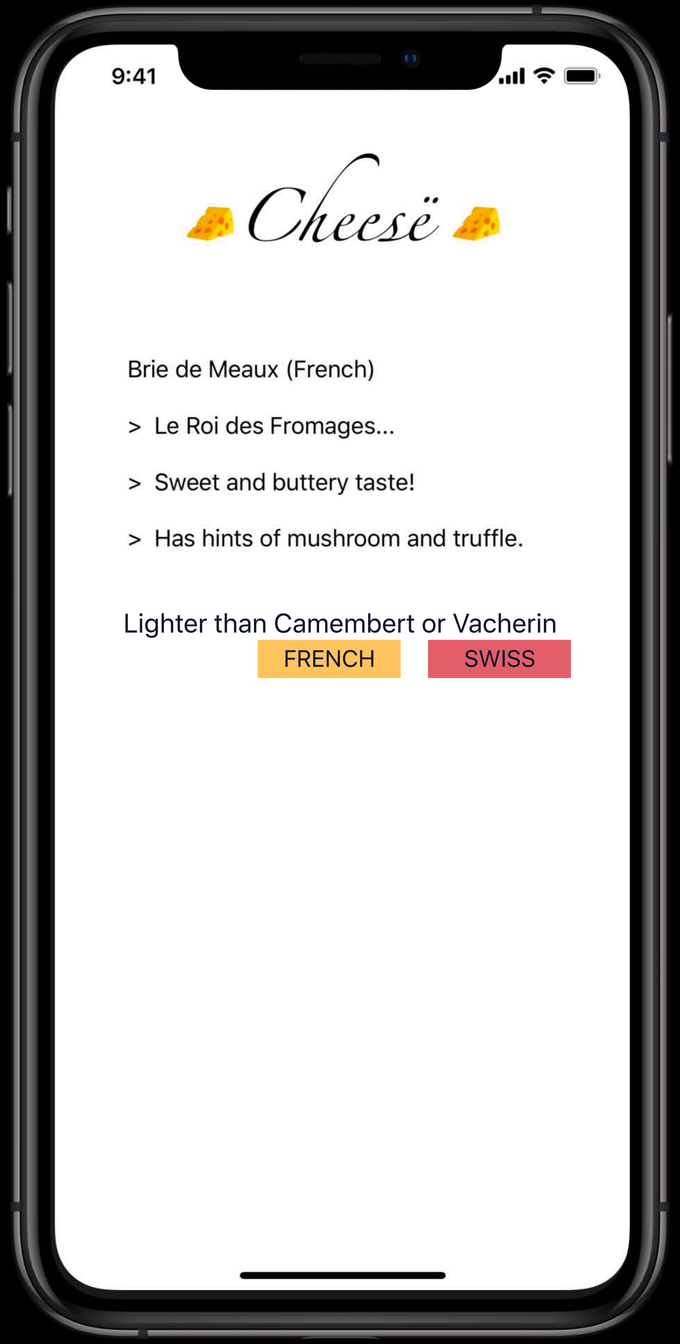
```
import NaturalLanguage

let gazetteer = try! NLGazetteer(contentsOf: url)

let tagger = NLTagger(tagSchemes: [.nameTypeOrLexicalClass])

tagger.setGazetteers([gazetteer], for: .nameTypeOrLexicalClass)
```

Text Catalog

Usage with a tagger

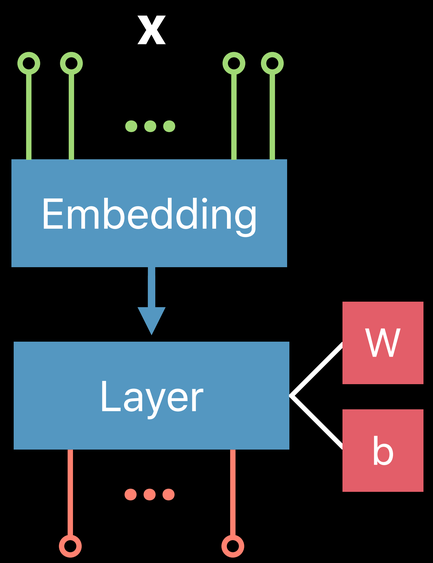
```
import NaturalLanguage

let gazetteer = try! NLGazetteer(contentsOf: url)

let tagger = NLTagger(tagSchemes: [.nameTypeOrLexicalClass])

tagger.setGazetteers([gazetteer], for: .nameTypeOrLexicalClass)
```


9:41
Cheesë
Brie de Meaux (French)
> Le Roi des Fromages...
> Sweet and buttery taste!
> Has hints of mushroom and truffle.
Lighter than Camembert or Vacherin


9:41
Cheesë
Brie de Meaux (French)
> Le Roi des Fromages...
> Sweet and buttery taste!
> Has hints of mushroom and truffle.
Lighter than Camembert or Vacherin
FRENCH
SWISS

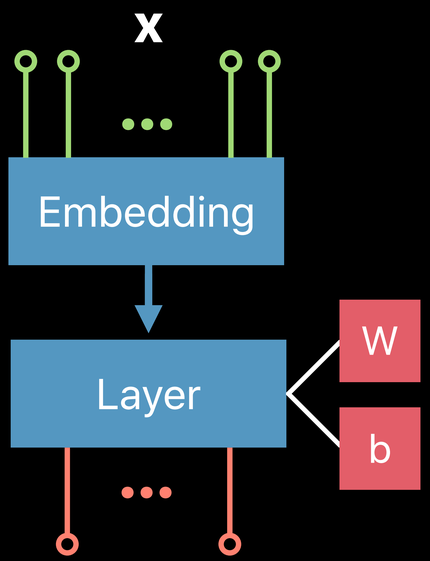
Word Embedding

Word Embedding



Neural Networks

Word Embedding



Neural Networks

Word Embeddings

Embedding

Concept

Embedding

Concept

Embedding

Concept

Mapping

Embedding

Concept

Mapping

	0.23	0.32	-0.45
	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
	0.81	0.05	-0.88

Vector Representation

Embedding

Concept

	0.23	0.32	-0.45
	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
	0.81	0.05	-0.88

Vector Representation

Embedding

Concept

	0.23	0.32	-0.45
	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
	0.81	0.05	-0.88

Vector Representation

Similarity in Vector Space

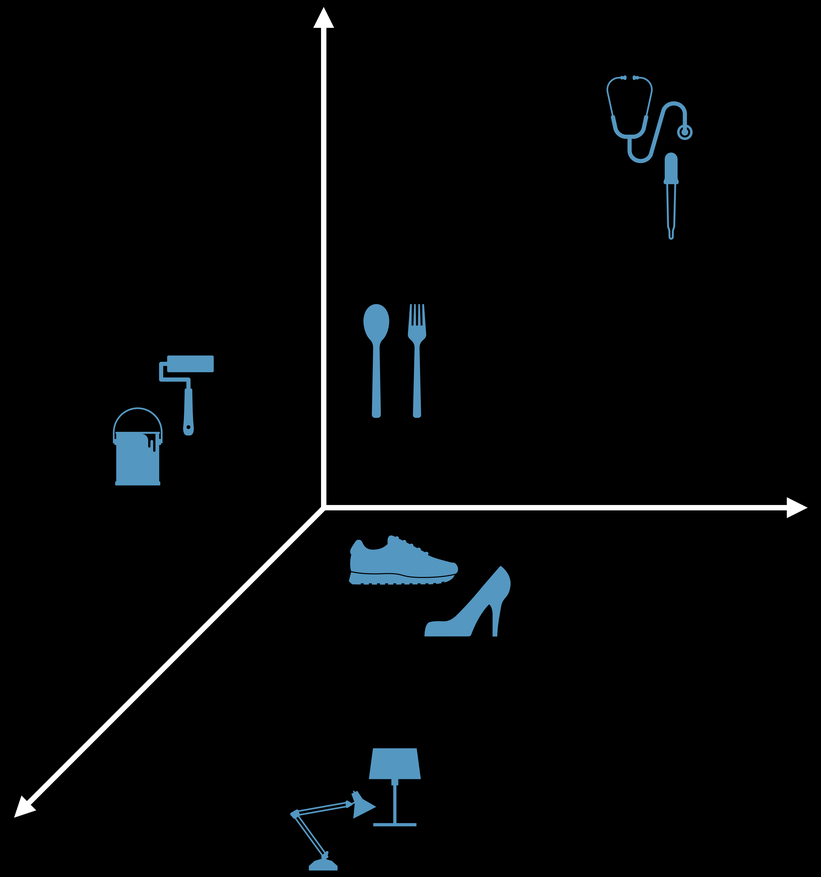
Embedding

Concept

	0.23	0.32	-0.45
	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
	0.81	0.05	-0.88

Vector Representation

Similarity in Vector Space



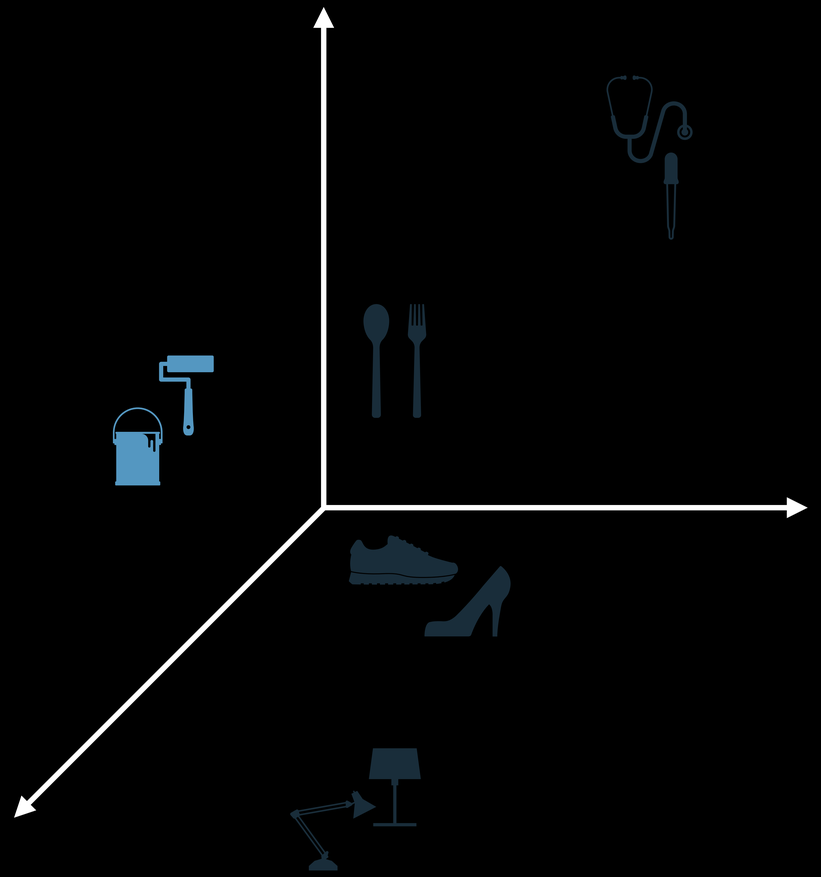
Embedding

Concept

	0.23	0.32	-0.45
	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
	0.81	0.05	-0.88

Vector Representation

Similarity in
Vector Space



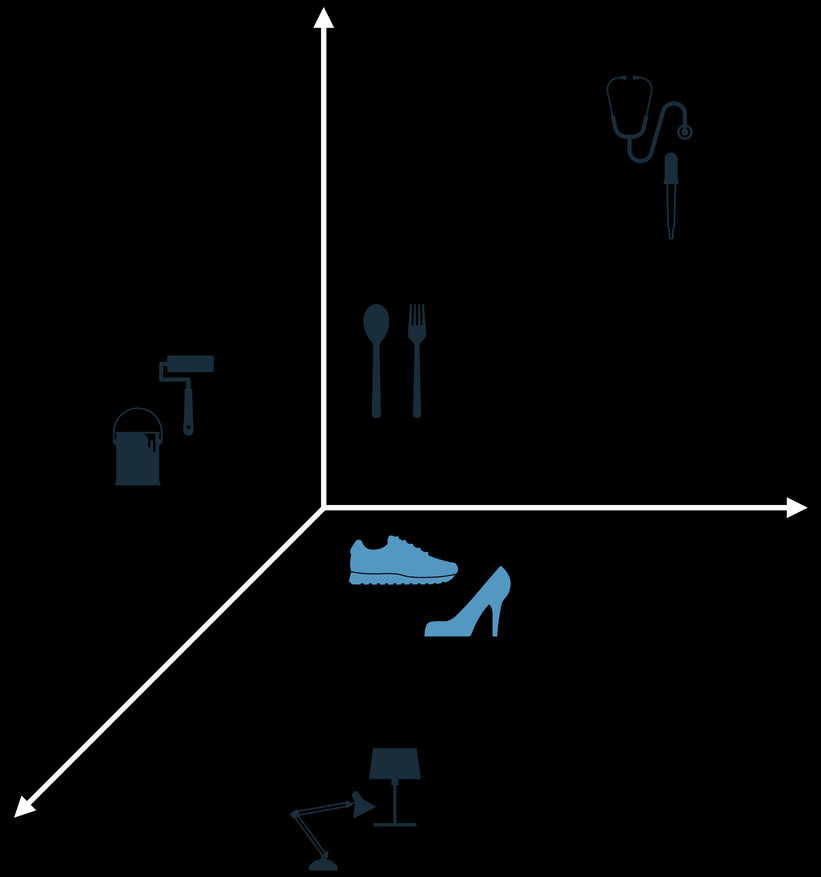
Embedding

Concept

	0.23	0.32	-0.45
	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
	0.81	0.05	-0.88

Vector Representation

Similarity in
Vector Space



Embedding

Types

	0.23	0.32	-0.45
	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
	0.81	0.05	-0.88

Vector Representation

Embedding

Types

Images

Words

Phrases

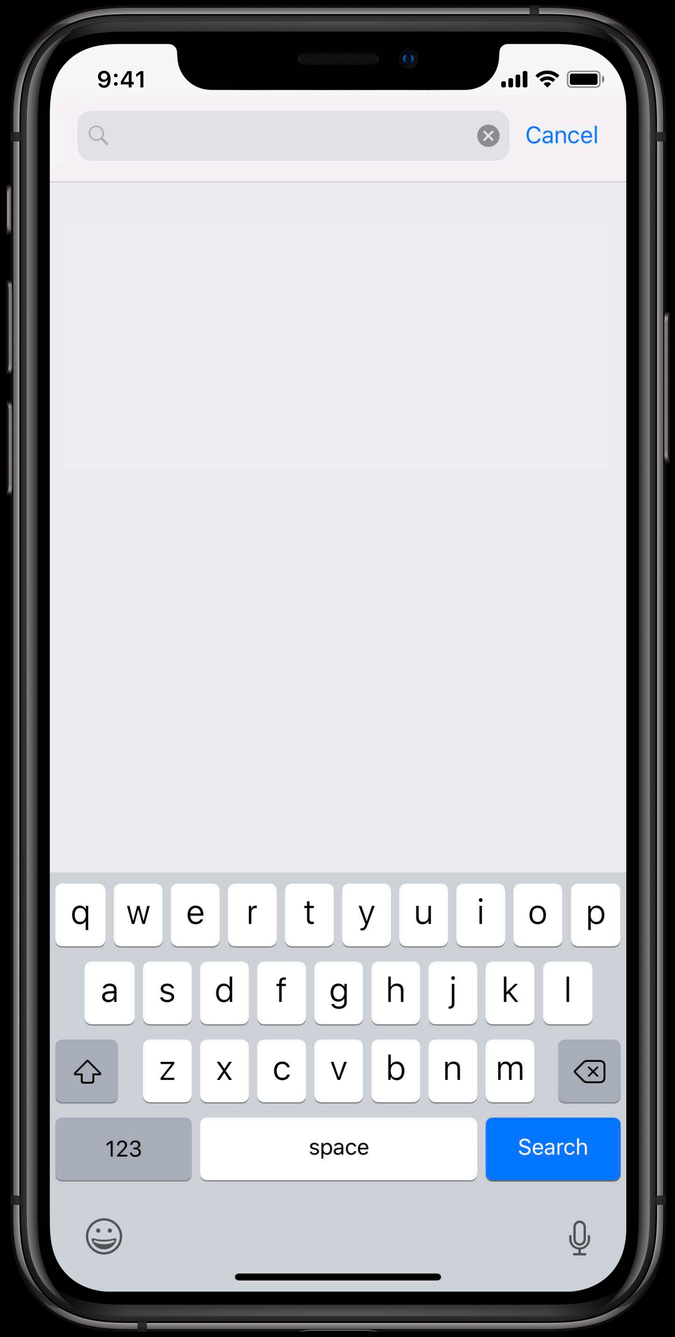
Song titles

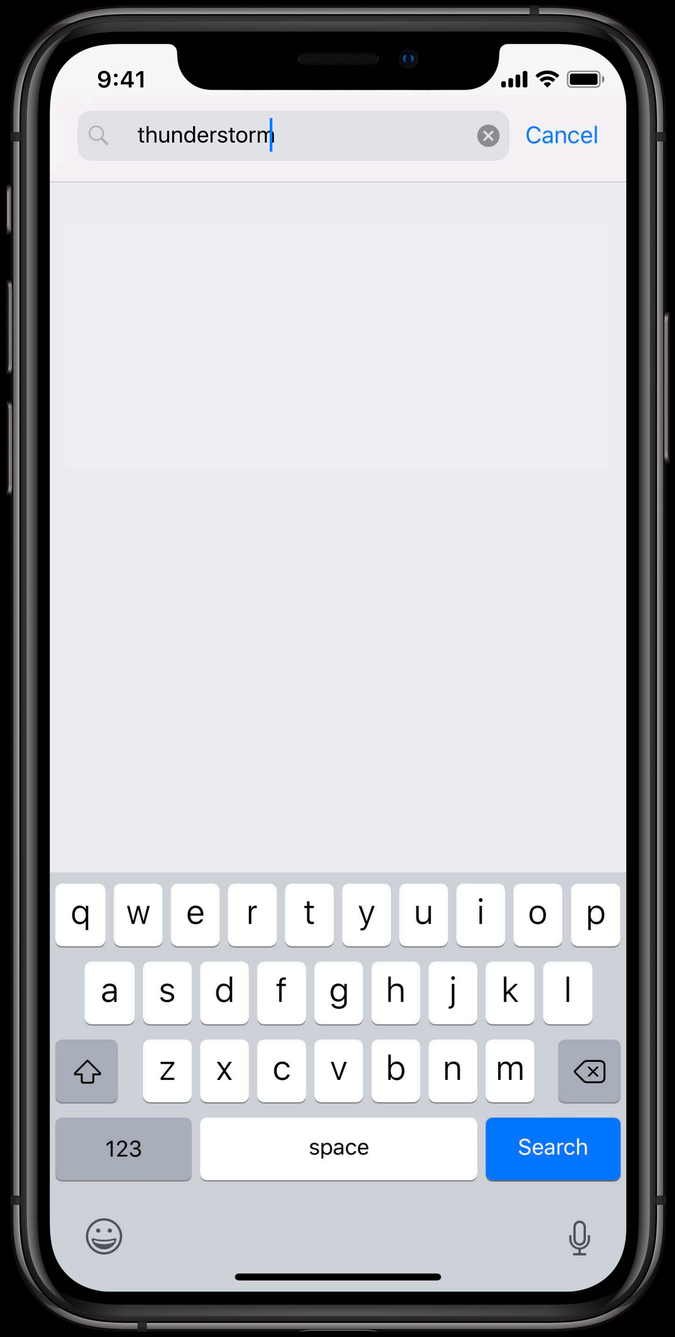
Product names

Mapping

string1	0.23	0.32	-0.45
string2	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
stringN	0.81	0.05	-0.88

Vector Representation


9:41
Cancel
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
Search


9:41
thunderstorm
Cancel
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
Search


9:41
thunderstorm
Cancel
Suggested Searches
thunderstorm → Sky
thunderstorm → Cloudy
14 Photos
See All
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
Search

Embedding

Get vector for word

Compute distance between two words

Get nearest neighbors for word

Get nearest neighbors for vector

OS Embedding

English

Spanish

French

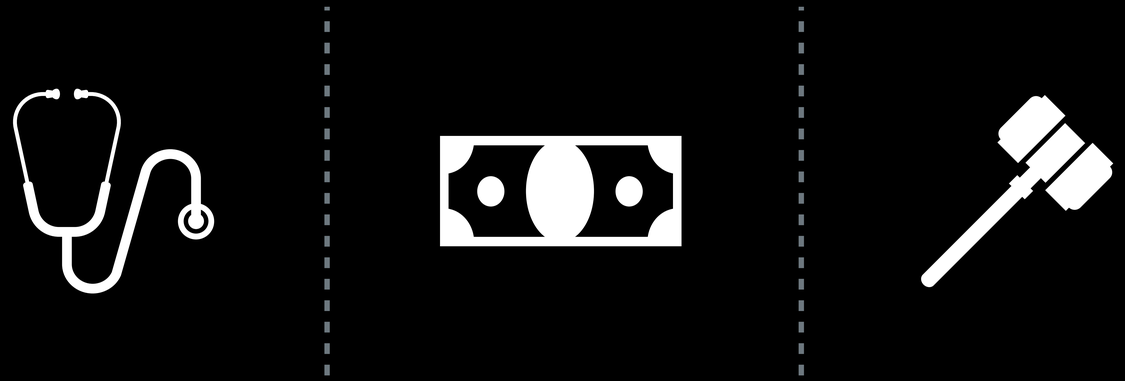
Italian

German

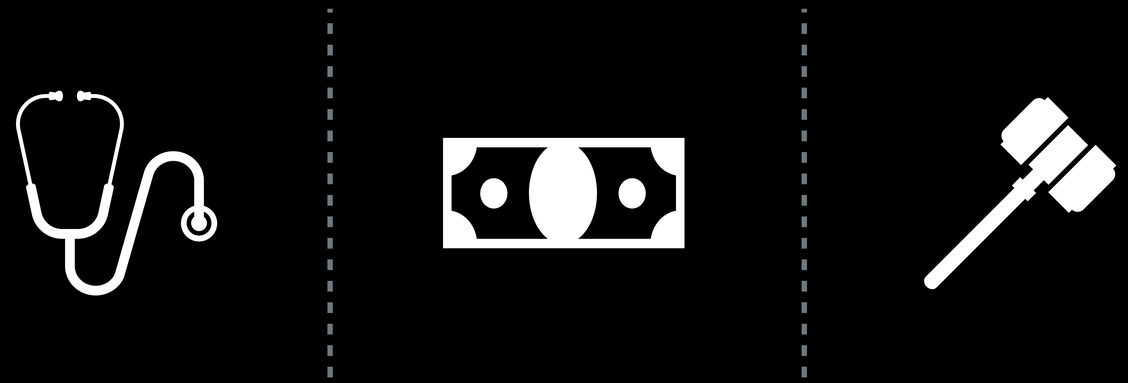
Portuguese

Simplified Chinese

Custom Word Embeddings



Custom Word Embeddings



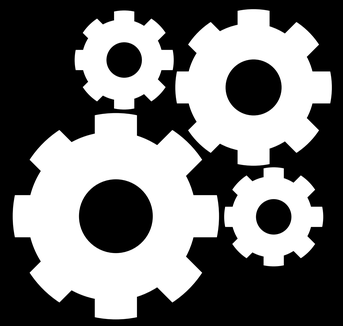
Domains

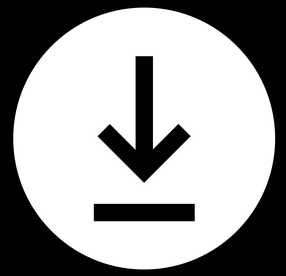
Vocabulary

Languages

Custom Word Embeddings

word2vec GloVe fasttext Custom Neural Network





Demo

Word Embeddings

Doug Davidson, NLP Technologies

Embedding

Obtain Embedding

```
import NaturalLanguage

guard let embedding = NLEmbedding.wordEmbedding(for: .english) else { return }

guard let vector = embedding.vector(for: string) else { return }

let distance = embedding.distance(between: word1, and: word2)

embedding.enumerateNeighbors(for: string, maximumCount: 5) { (string, distance) -> Bool in
  // make use of string and distance
  return true
}
```

Embedding

Obtain Embedding

```
import NaturalLanguage

guard let embedding = NLEmbedding.wordEmbedding(for: .english) else { return }

guard let vector = embedding.vector(for: string) else { return }

let distance = embedding.distance(between: word1, and: word2)

embedding.enumerateNeighbors(for: string, maximumCount: 5) { (string, distance) -> Bool in
  // make use of string and distance
  return true
}
```

Embedding

Get vector

```
import NaturalLanguage

guard let embedding = NLEmbedding.wordEmbedding(for: .english) else { return }

guard let vector = embedding.vector(for: string) else { return }

let distance = embedding.distance(between: word1, and: word2)

embedding.enumerateNeighbors(for: string, maximumCount: 5) { (string, distance) -> Bool in
  // make use of string and distance
  return true
}
```

Embedding

Get vector

```
import NaturalLanguage

guard let embedding = NLEmbedding.wordEmbedding(for: .english) else { return }

guard let vector = embedding.vector(for: string) else { return }

let distance = embedding.distance(between: word1, and: word2)

embedding.enumerateNeighbors(for: string, maximumCount: 5) { (string, distance) -> Bool in
  // make use of string and distance
  return true
}
```

Embedding

Distance between words

```
import NaturalLanguage

guard let embedding = NLEmbedding.wordEmbedding(for: .english) else { return }

guard let vector = embedding.vector(for: string) else { return }

let distance = embedding.distance(between: word1, and: word2)

embedding.enumerateNeighbors(for: string, maximumCount: 5) { (string, distance) -> Bool in
  // make use of string and distance
  return true
}
```

Embedding

Distance between words

```
import NaturalLanguage

guard let embedding = NLEmbedding.wordEmbedding(for: .english) else { return }

guard let vector = embedding.vector(for: string) else { return }

let distance = embedding.distance(between: word1, and: word2)

embedding.enumerateNeighbors(for: string, maximumCount: 5) { (string, distance) -> Bool in
  // make use of string and distance
  return true
}
```

Embedding

Nearest neighbors

```
import NaturalLanguage

guard let embedding = NLEmbedding.wordEmbedding(for: .english) else { return }

guard let vector = embedding.vector(for: string) else { return }

let distance = embedding.distance(between: word1, and: word2)

embedding.enumerateNeighbors(for: string, maximumCount: 5) { (string, distance) -> Bool in
  // make use of string and distance
  return true
}
```

Embedding

Nearest neighbors

```
import NaturalLanguage

guard let embedding = NLEmbedding.wordEmbedding(for: .english) else { return }

guard let vector = embedding.vector(for: string) else { return }

let distance = embedding.distance(between: word1, and: word2)

embedding.enumerateNeighbors(for: string, maximumCount: 5) { (string, distance) -> Bool in
  // make use of string and distance
  return true
}
```

Custom Word Embeddings

```
import CreateML

let vectors = ["Camembert":  [0.118, 0.013, -0.063, -0.020, -0.100, 0.088, …],
               "Brie": [0.038, 0.008, -0.051,  0.065, -0.198, 0.024, …],
               "Cirrus": [0.128, 0.127,  0.021, -0.042, -0.057, 0.055, …],
               "Neufchâtel": [0.308, 0.094, -0.011,  0.155, -0.005, 0.021…], …]

let embedding = try MLWordEmbedding(dictionary: vectors)

try embedding.write(to: url)
```

Custom Word Embeddings

```
import CreateML

let vectors = ["Camembert":  [0.118, 0.013, -0.063, -0.020, -0.100, 0.088, …],
               "Brie": [0.038, 0.008, -0.051,  0.065, -0.198, 0.024, …],
               "Cirrus": [0.128, 0.127,  0.021, -0.042, -0.057, 0.055, …],
               "Neufchâtel": [0.308, 0.094, -0.011,  0.155, -0.005, 0.021…], …]

let embedding = try MLWordEmbedding(dictionary: vectors)

try embedding.write(to: url)
```

Custom Word Embeddings

```
import CreateML

let vectors = ["Camembert":  [0.118, 0.013, -0.063, -0.020, -0.100, 0.088, …],
               "Brie": [0.038, 0.008, -0.051,  0.065, -0.198, 0.024, …],
               "Cirrus": [0.128, 0.127,  0.021, -0.042, -0.057, 0.055, …],
               "Neufchâtel": [0.308, 0.094, -0.011,  0.155, -0.005, 0.021…], …]

let embedding = try MLWordEmbedding(dictionary: vectors)

try embedding.write(to: url)
```

Custom Word Embeddings

```
import CreateML

let vectors = ["Camembert":  [0.118, 0.013, -0.063, -0.020, -0.100, 0.088, …],
               "Brie": [0.038, 0.008, -0.051,  0.065, -0.198, 0.024, …],
               "Cirrus": [0.128, 0.127,  0.021, -0.042, -0.057, 0.055, …],
               "Neufchâtel": [0.308, 0.094, -0.011,  0.155, -0.005, 0.021…], …]

let embedding = try MLWordEmbedding(dictionary: vectors)

try embedding.write(to: url)
```

Custom Word Embeddings

```
import CreateML

let vectors = ["Camembert":  [0.118, 0.013, -0.063, -0.020, -0.100, 0.088, …],
               "Brie": [0.038, 0.008, -0.051,  0.065, -0.198, 0.024, …],
               "Cirrus": [0.128, 0.127,  0.021, -0.042, -0.057, 0.055, …],
               "Neufchâtel": [0.308, 0.094, -0.011,  0.155, -0.005, 0.021…], …]

let embedding = try MLWordEmbedding(dictionary: vectors)

try embedding.write(to: url)
```

Custom Word Embeddings

```
import CreateML

let vectors = ["Camembert":  [0.118, 0.013, -0.063, -0.020, -0.100, 0.088, …],
               "Brie": [0.038, 0.008, -0.051,  0.065, -0.198, 0.024, …],
               "Cirrus": [0.128, 0.127,  0.021, -0.042, -0.057, 0.055, …],
               "Neufchâtel": [0.308, 0.094, -0.011,  0.155, -0.005, 0.021…], …]

let embedding = try MLWordEmbedding(dictionary: vectors)

try embedding.write(to: url)
```

Under the Hood

- Automatic compression
- Efficient representation for fast k-NN

Under the Hood

Automatic compression

Efficient representation for fast k-NN

Embedding	Vocabulary	Dimension	Size (MB)	20-NN time (ms)
GloVe	400,000	300	31	2.6
fastText	1,000,000	300	56	1.5

Podcast Embeddings

Podcast Embeddings

Recommender system embeddings

Podcast Embeddings

Recommender system embeddings

Daily Therapy

The Art of Living

Football:
Euro League

...

Game Talk:
Know Your Game Apps

Mapping

string1	0.23	0.32	-0.45
string2	0.28	0.40	-0.39
...	...	...	...
	0.75	0.02	-0.95
stringN	0.81	0.05	-0.88

Vector Representation

Podcast Embeddings

Podcast Embeddings

66K podcast embeddings from Apple Media

Podcast Embeddings

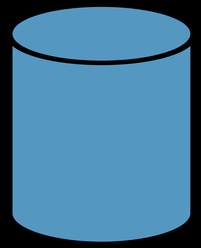
66K podcast embeddings from Apple Media

Transfer Learning

Text Classification

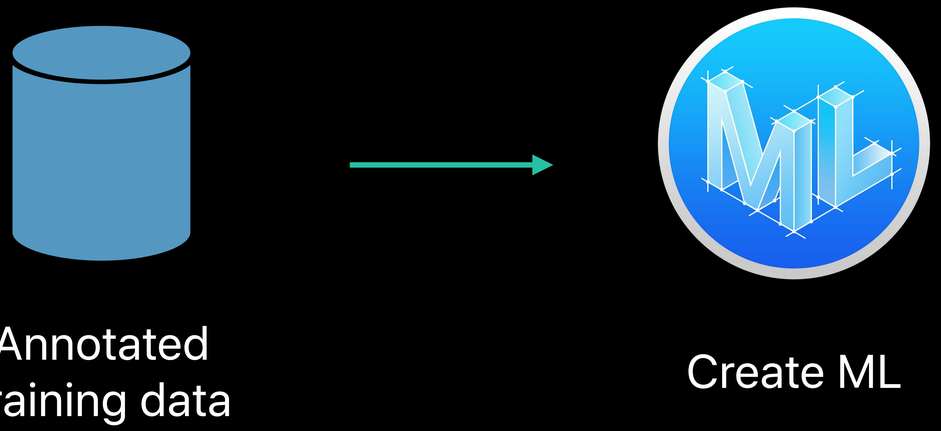
Custom Text Classification

Custom Text Classification



Annotated
training data

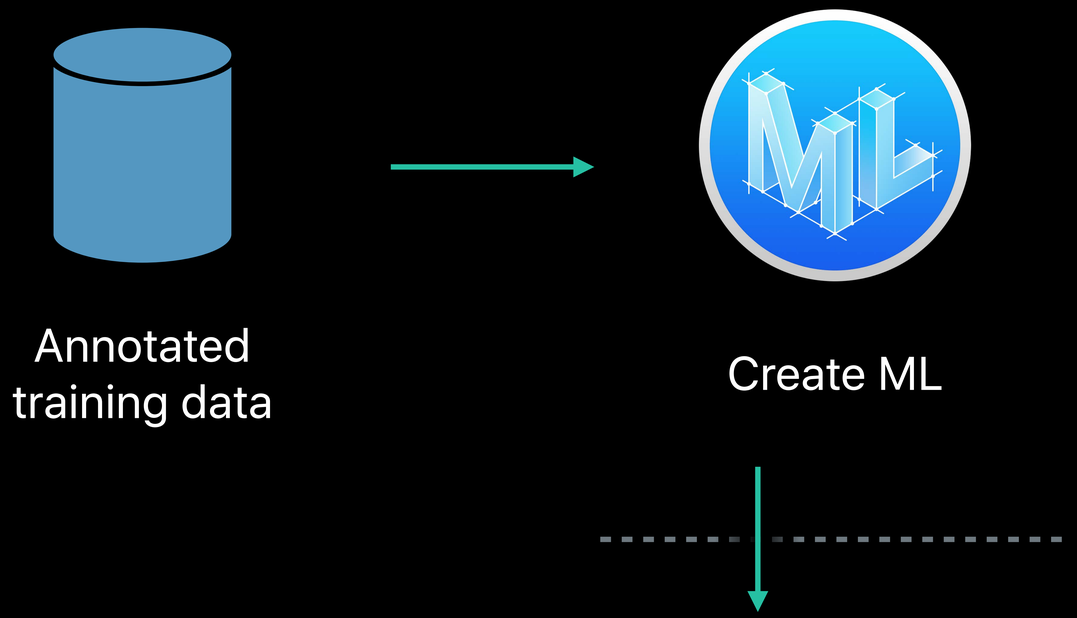
Custom Text Classification



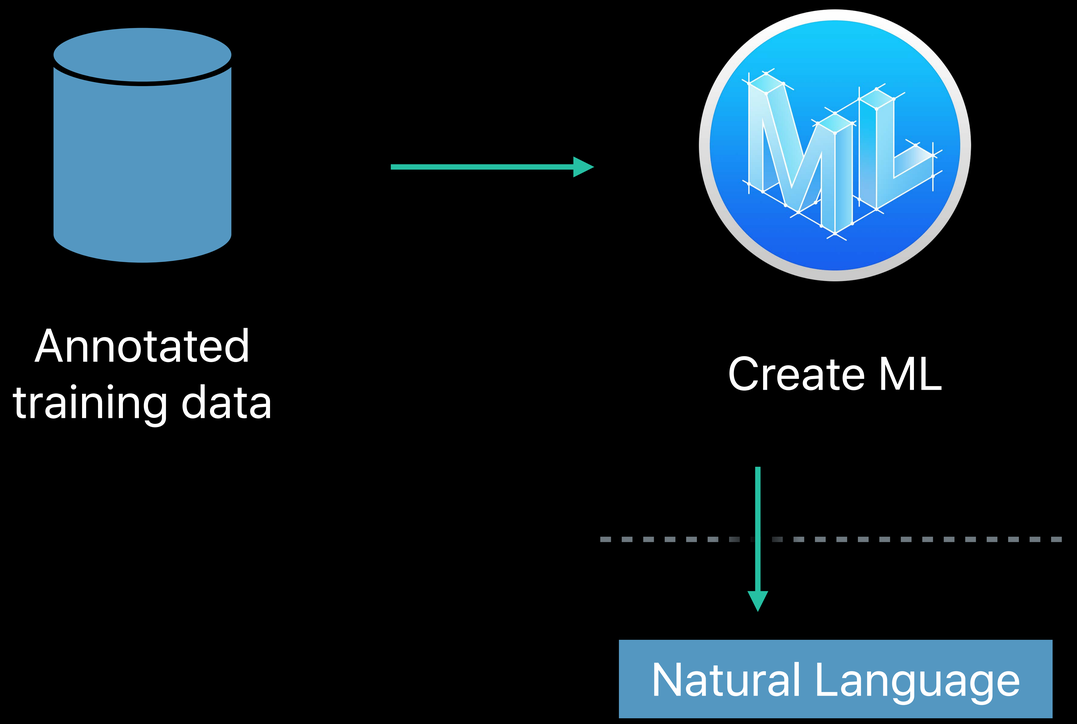
Annotated
training data

Create ML

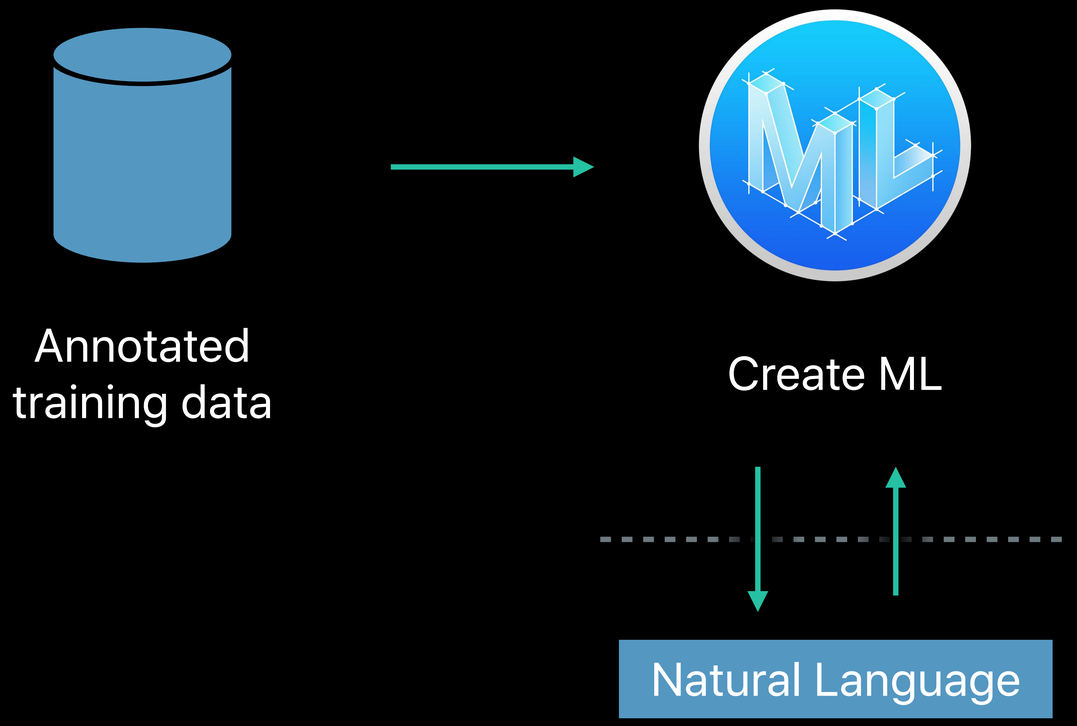
Custom Text Classification



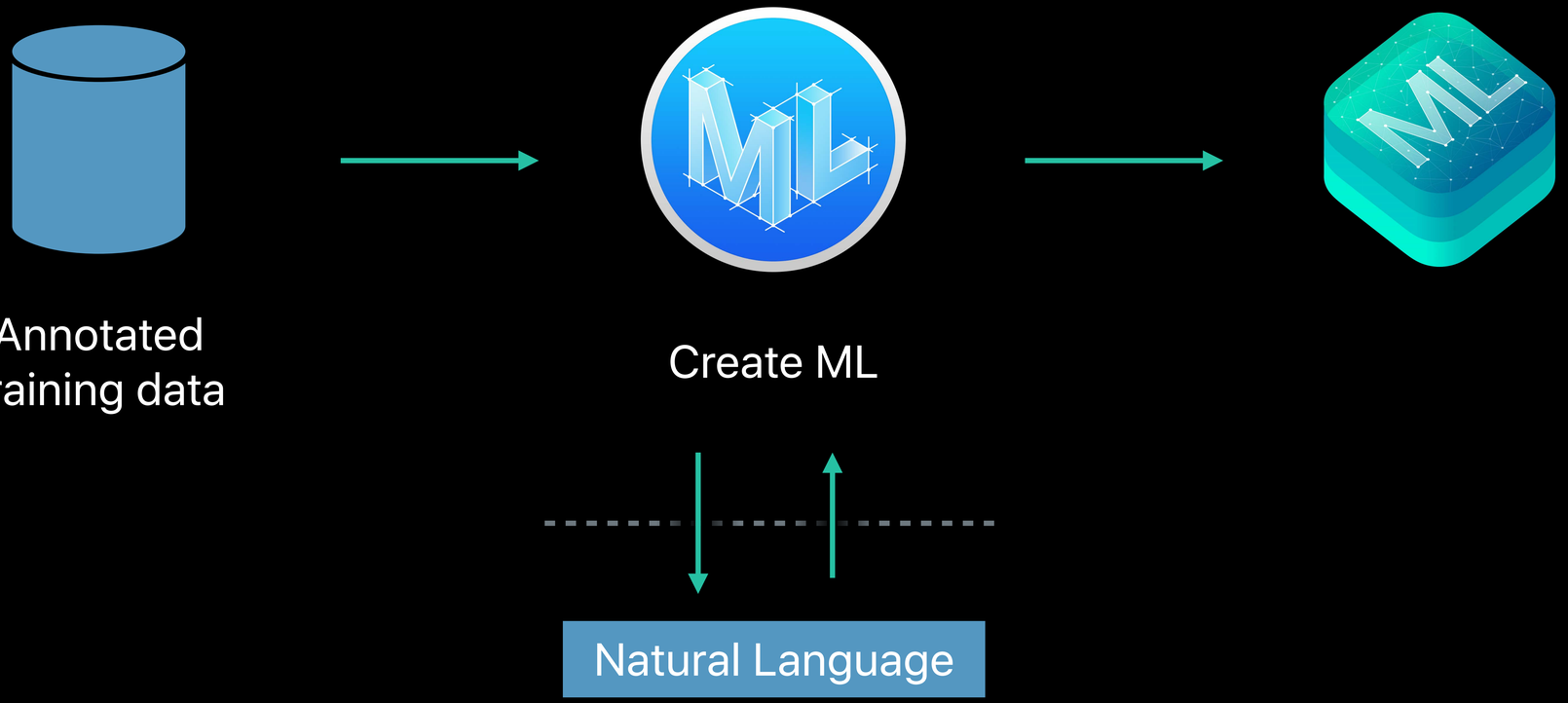
Custom Text Classification



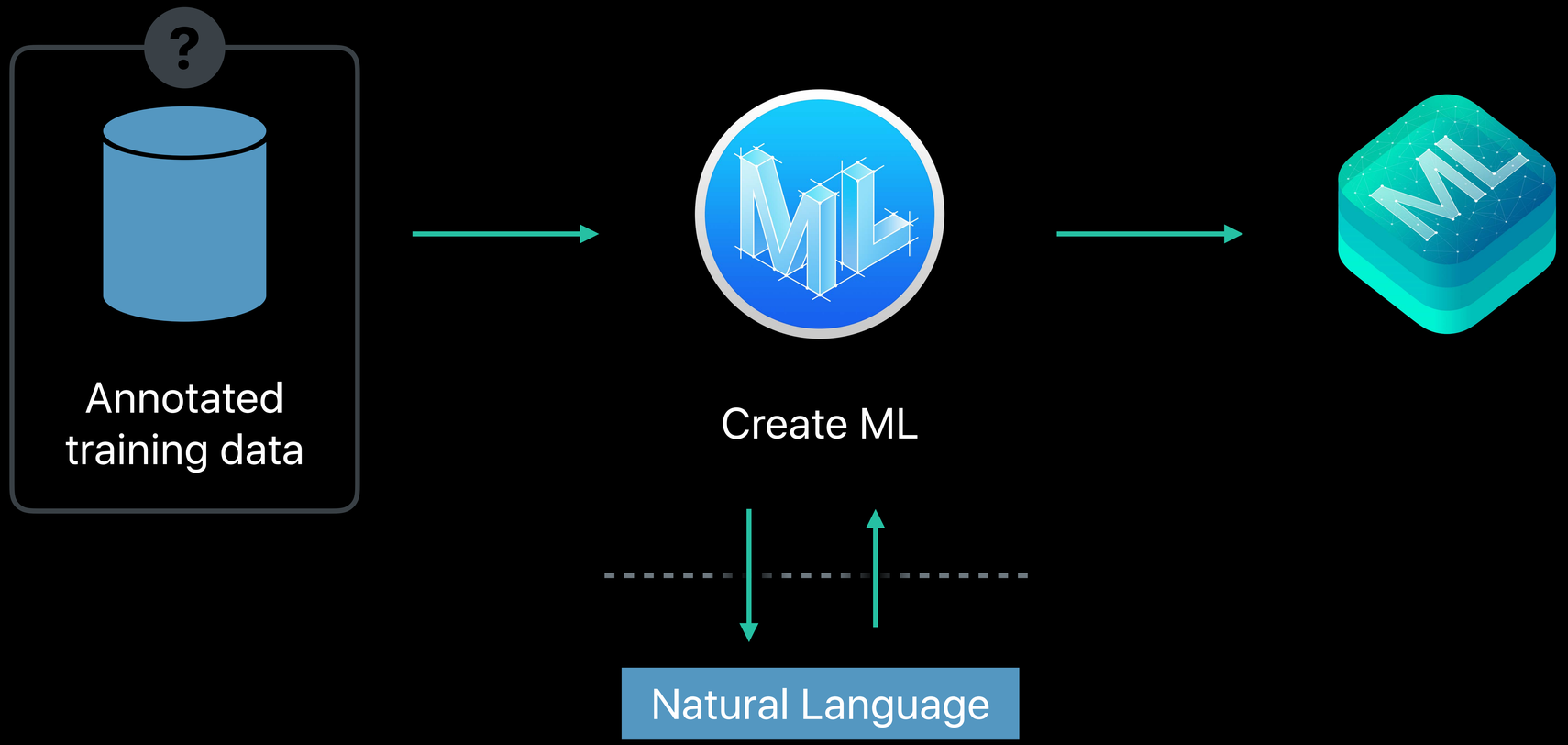
Custom Text Classification



Custom Text Classification



Custom Text Classification



Transfer Learning

Text Classification

Smaller annotated
training data

Transfer Learning

Text Classification

+

Smaller annotated training data

Pre-trained model

Transfer Learning

Text Classification

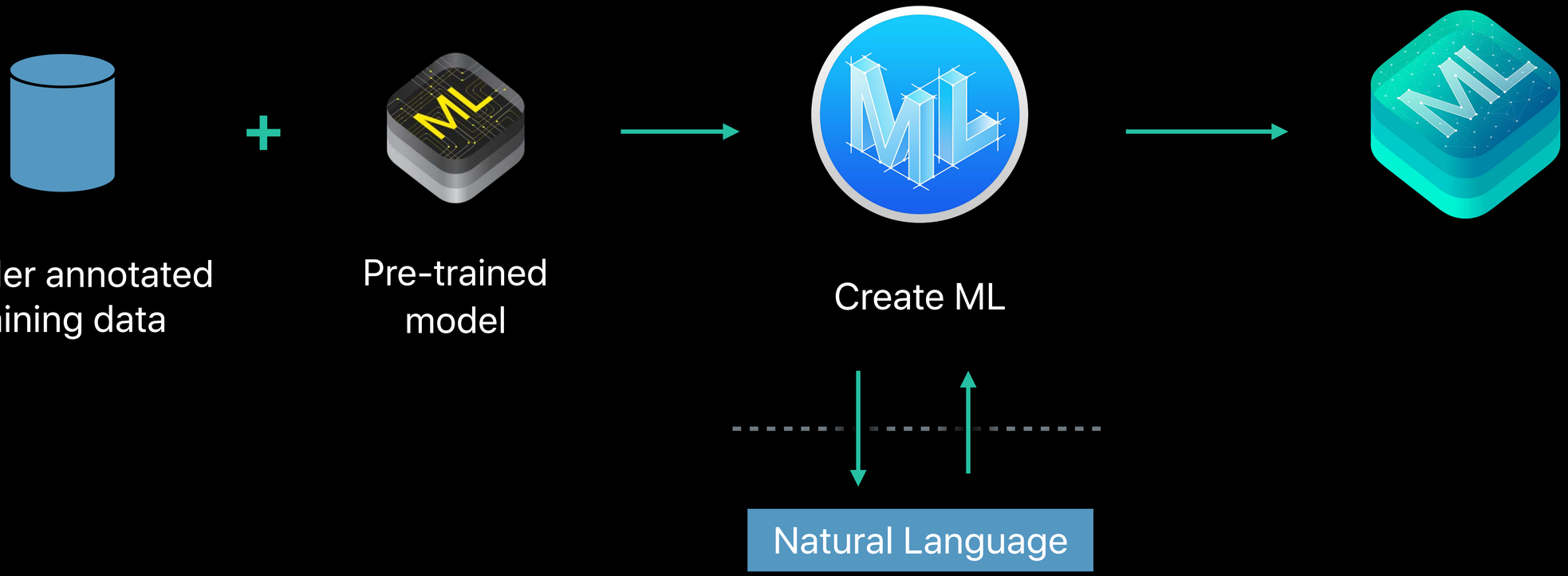
Smaller annotated training data

Pre-trained model

Create ML

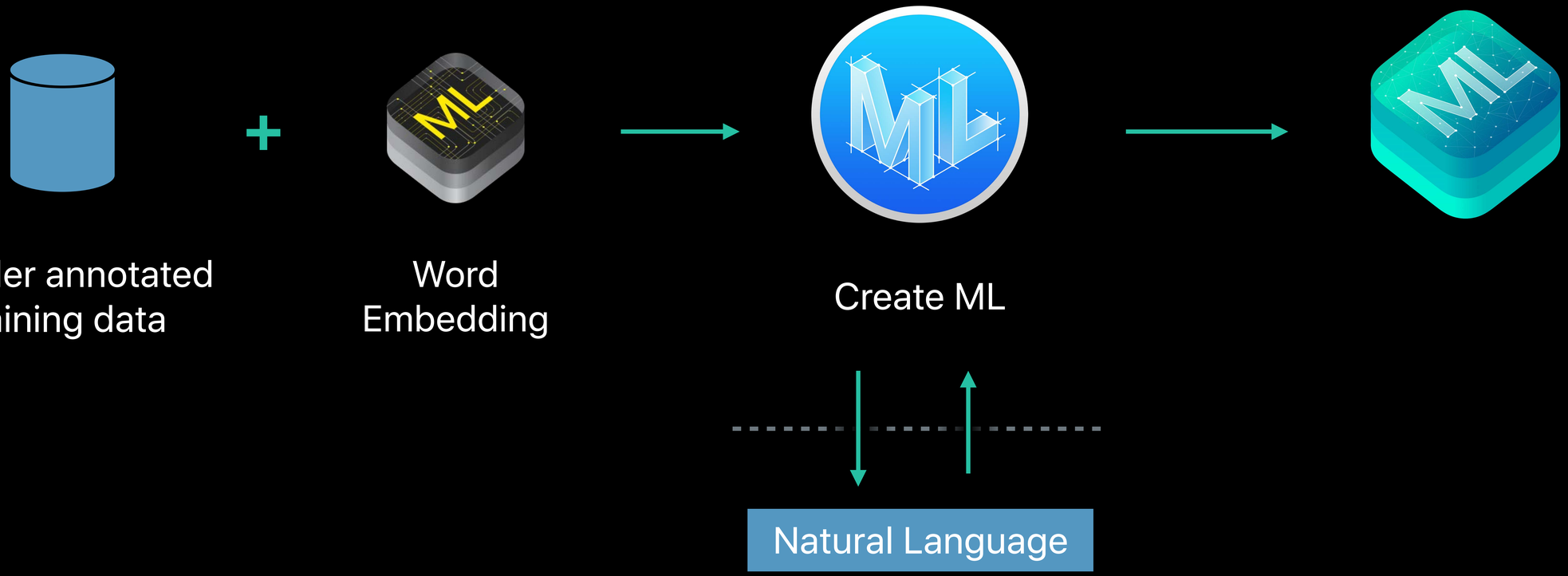
Transfer Learning

Text Classification



Transfer Learning

Text Classification



Transfer Learning

Text Classification

```
import CreateML

let modelParameters =
MLTextClassifier.ModelParameters(algorithm: .transferLearning(version: 1))
```

Transfer Learning

Text Classification

```
import CreateML

let modelParameters =
MLTextClassifier.ModelParameters(algorithm: .transferLearning(version: 1))
```

Transfer Learning

Text Classification

```
import CreateML

let modelParameters =
MLTextClassifier.ModelParameters(algorithm: .transferLearning(.staticEmbedding, version: 1))
```

OS Embeddings

Transfer Learning

Text Classification

```
import CreateML

let modelParameters =
MLTextClassifier.ModelParameters(algorithm: .transferLearning(.staticEmbedding, version: 1))
```

OS Embeddings

Transfer Learning

Text Classification

```
import CreateML

let modelParameters =
MLTextClassifier.ModelParameters(algorithm: .transferLearning(.customEmbedding(url), version: 1))
```

Custom Compressed Embeddings

Transfer Learning

Text Classification

```
import CreateML

let modelParameters =
MLTextClassifier.ModelParameters(algorithm: .transferLearning(.customEmbedding(url), version: 1))
```

Custom Compressed Embeddings

Transfer Learning

Embeddings

USA leads in *apple* production

iPhone was introduced by *Apple* in 2007

Transfer Learning

Embeddings

USA leads in *apple* production

iPhone was introduced by *Apple* in 2007

apple	-0.83	0.32	0.48	-0.89	0.41	0.78	...	...	0.68

Transfer Learning

Embeddings

Transfer Learning

Text Classification

```
import CreateML

let modelParameters =
MLTextClassifier.ModelParameters(algorithm: .transferLearning(.dynamicEmbedding, version: 1))
```

Dynamic Embedding
using Neural Network

Transfer Learning

Text Classification

```
import CreateML

let modelParameters =
MLTextClassifier.ModelParameters(algorithm: .transferLearning(.dynamicEmbedding, version: 1))
```

Dynamic Embedding
using Neural Network

Demo

Transfer Learning for Text Classification

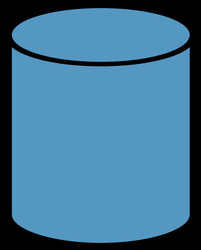
Supported Languages

Transfer Learning

Static Embedding	English French Italian German Spanish Portuguese Simplified Chinese
Dynamic Embedding	English Spanish

Guidelines

Data



Understand target domain

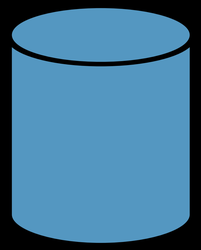
Prepare representative data

Cover variation expected

Create exclusive data splits of sufficient size

Guidelines

Data



Understand target domain

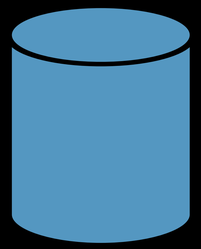
Prepare representative data

Cover variation expected

Create exclusive data splits of sufficient size

Guidelines

Data



Understand target domain

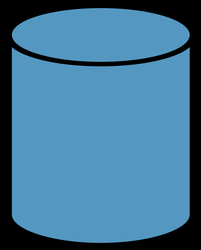
Prepare representative data

Cover variation expected

Create exclusive data splits of sufficient size

Guidelines

Data



Understand target domain

Prepare representative data

Cover variation expected

Create exclusive data splits of sufficient size

Guidelines

Algorithms

Guidelines

Algorithms

Start with maxEnt classifier

Guidelines

Algorithms

Start with maxEnt classifier

Training

I love ice cream…can’t wait for summer	+
I hate cold weather	-
We are really happy with our new home	+
We are extremely unhappy with the insurance company	-

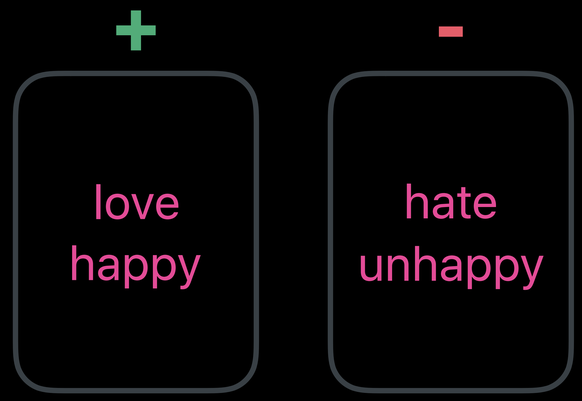
Guidelines

Algorithms

Start with maxEnt classifier

Training

I love ice cream…can’t wait for summer	+
I hate cold weather	-
We are really happy with our new home	+
We are extremely unhappy with the insurance company	-



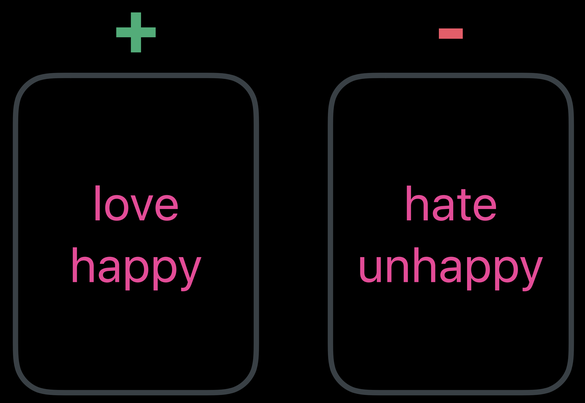
Guidelines

Algorithms

Start with maxEnt classifier

Inference

I love mystery novels	+
I hate driving on rainy days	-
I'm so happy that mom got me a scooter	+
She was really unhappy when she spoke with me	-



Guidelines

Algorithms

Start with maxEnt classifier

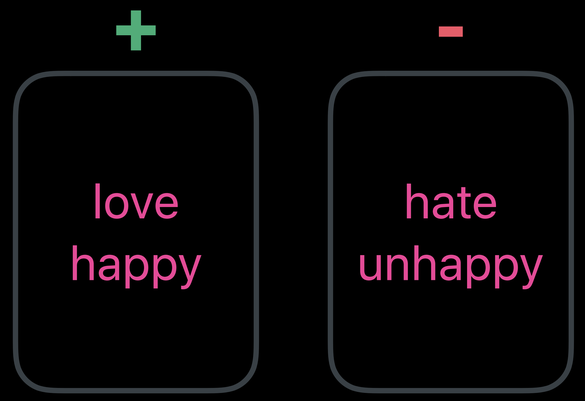
Guidelines

Algorithms

Start with maxEnt classifier

Inference

I enjoy mystery novels	+
I despise driving on rainy days	-



Guidelines

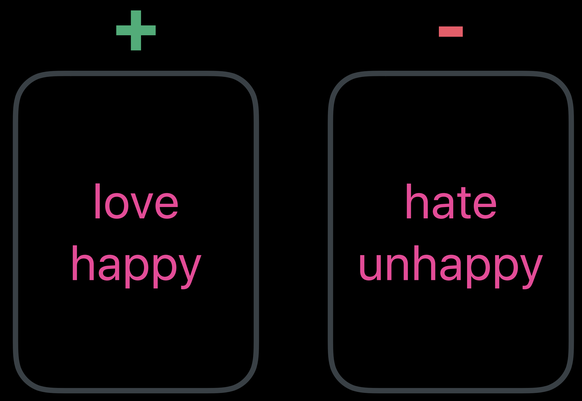
Algorithms

Start with maxEnt classifier

Compare with transfer learning

Inference

I enjoy mystery novels	+
I despise driving on rainy days	-



Summary

Sentiment Analysis

Text Catalogs

Word Embeddings

Transfer Learning for Text Classification

More Information

developer.apple.com/wwdc19/232

What’s New in Machine Learning	WWDC 2019
Create ML for Activity, Text and Recommendations	WWDC 2019


WWDC19