ACADEMY OF Management

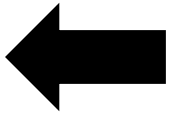
Translating Text into Meaning: Decoding Natural Language with Embeddings

Ruixiang Song

rsong@lsu.edu

ACADEMY OF Management

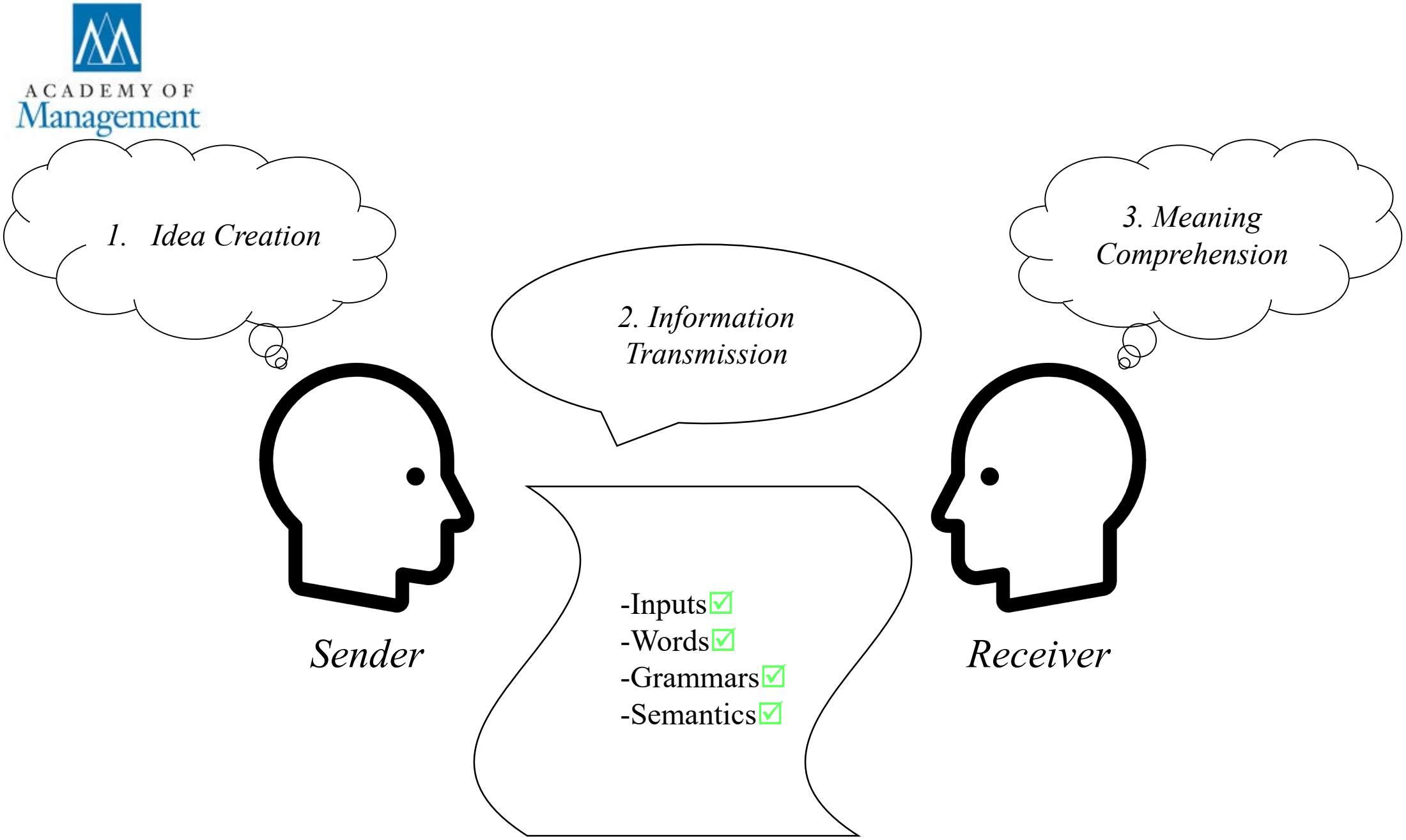
1. **Data Collection and Preprocessing**
2. **Model Training**
3. **Evaluation and Optimization**
4. **Application**



An embedding is a numerical map of a piece of information (*text, documents, images, audio*…)

The representation captures the semantic meaning of what is being embedded, making it robust for many applications.

The process of embedding typically involves translating high-dimensional information into low-dimensional data.


ACADEMY OF
Management
1. Idea Creation
2. Information Transmission
3. Meaning Comprehension
Sender
-Inputs☑
-Words☑
-Grammars☑
-Semantics☑
Receiver

ACADEMY OF Management

1. *Idea Creation*

2. *[illegible]*

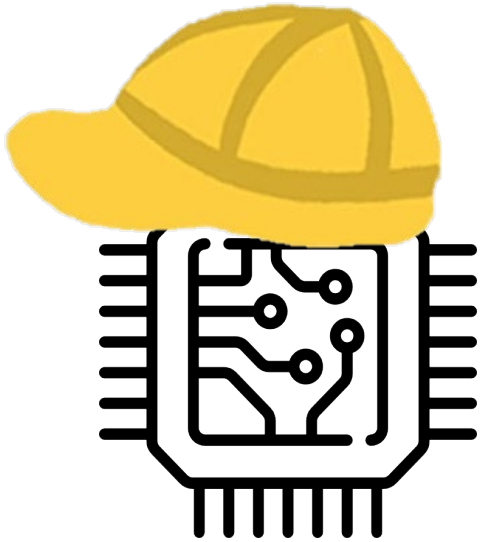
3. *[illegible]*

Sender

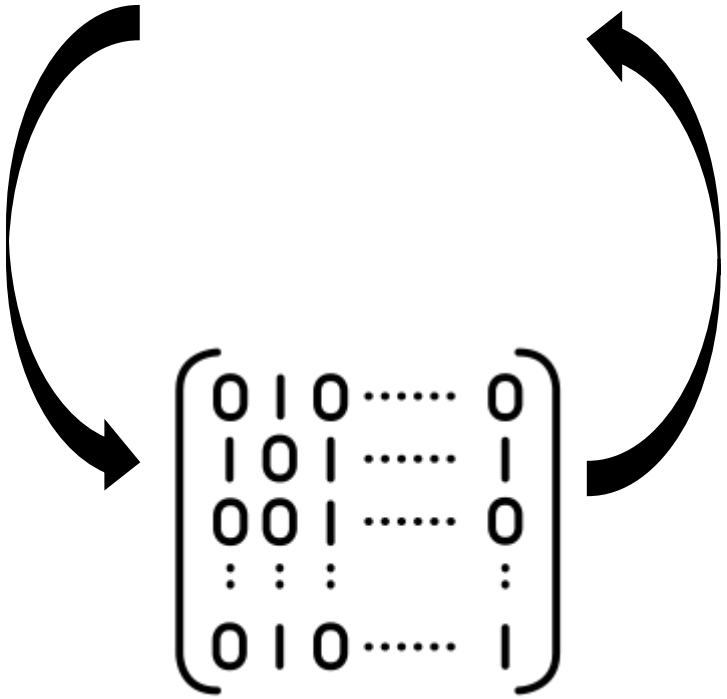
-Inputs ☑
-Words ☒
-Grammars ☒
-Semantics ☒

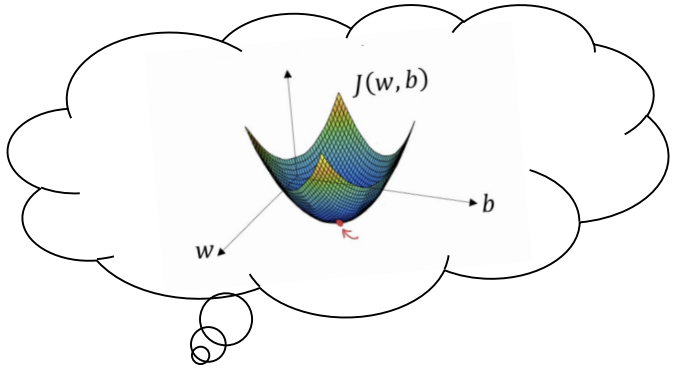
Receiver

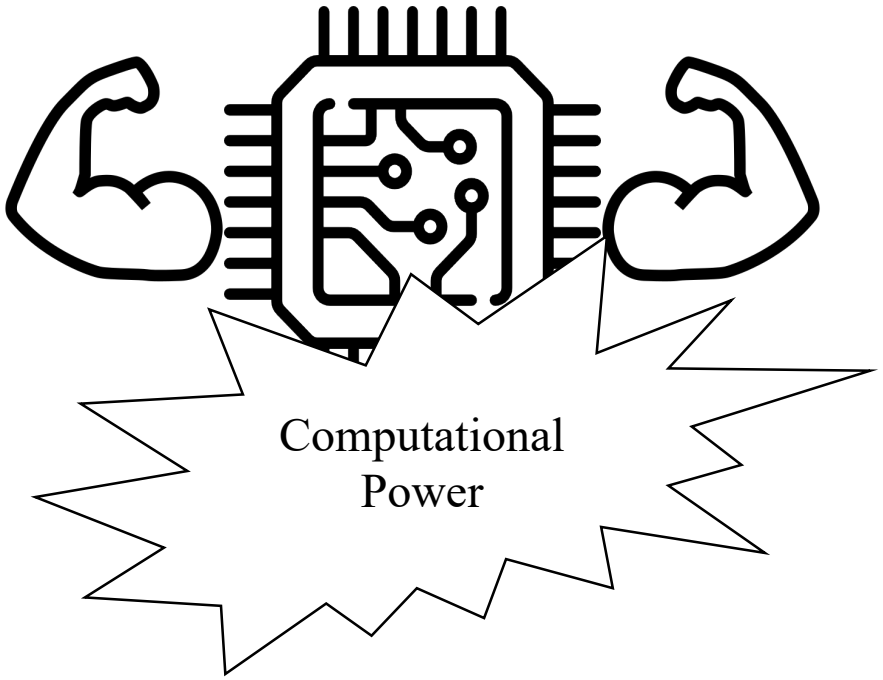
ACADEMY OF Management



"Hello, World!"







ACADEMY OF Management

Morse code: *a simple and somewhat primitive form of embedding*

Library catalog: *a catalog system transforms the collection of books into a structured format that can be easily searched and utilized.*

ACADEMY OF Management

pension, mutual, bank, & pension and mutual

⬇

pension	[1, 0, 0, 0]
mutual	[0, 1, 0, 0]
bank	[0, 0, 1, 0]
pension and mutual	[0, 0, 0, 1] ☑
pension and mutual	[1, 1, 0, 0] ☒

1. “pension funds invest in us”
2. “pension funds and mutual funds invest in them”

⬇

	[pension, and, mutual, funds, invest, in, us, them]
1st	[1, 0, 0, 1, 1, 1, 1, 0]
2nd	[1, 1, 1, 2, 1, 1, 0, 1]

One-Hot Encoding: *Each word in a corpus is represented as a unique single-layer binary vector (bit array) in an n-dimensional space, where n is the number of unique words in the corpus. Each word is represented by a vector that has a '1' in the position corresponding to that word and '0's elsewhere*

Count Vectorization: *Each document in a corpus is represented as a single-layer non-binary vector in an n-dimensional space, where n is the number of unique words in the entire corpus. Each element in the vector is a count of the occurrences of the corresponding word in the document. This is also commonly referred to as* ***Bag of Words (BoW)****, especially when it focuses on only the presence of the word, rather than the count.*

1. “pension funds invest in us”
2. “pension funds and mutual funds invest in them”

[pension funds, mutual funds, invest in us]

1st [1, 0, 1]

2nd [1, 1, 0]

N-grams: *This method creates sequences of 'n' items from a given sample of text or speech. In the context of natural language processing, an n-gram is a contiguous sequence of n items from a given sample of text or speech. The 'items' in this case are usually words, but could also be characters, syllables, or other units depending on the application.*

1. “pension funds invest in us”
2. “pension funds and mutual funds invest in them”

[pension, and, mutual, funds, invest, in, us, them]

1st [0, 0, 0, 0, 0, 0, 0.1, 0]

2nd [0, 0.1, 0.1, 0, 0, 0, 0, 0.1]

TF-IDF (Term Frequency-Inverse Document Frequency): *This method adjusts the count vectorization by accounting for words that are common across all documents. The TF-IDF score for each word in each document is calculated.*

ACADEMY OF Management

- **Simple Statistical Embeddings**

- ***One-hot encoding***
- ***Bag-of-Words***
- ***Count Vectorization***
- ***N-grams***
- ***TF-IDF***

Basic Text Analyses

- ***Information extraction***
- ***Semantic Similarity***
- ***Text Matching***
- ***Sentiment Capturing***

ACADEMY OF Management

	Model Architecture	Computational Intensity	Result Interpretability	Context Understanding	Semantic Capturing	Generation Predictivity	Task Versatility
Word2Vec	*Two-layer NN*	Low	High	Low	Medium	Low	Low
GloVe	*Weighted-2LS*	Low	High	Low	Medium	Low	Low
ELMo	*Two-layer LSTM*	Medium	Medium	Medium	Medium	Medium	Medium
BERT	*Multiple transformer layers (110m)*	Medium to High	Low	High	High	Medium	High
GPT	*Multiple transformer layers(175b)*	Medium to High	Low	Medium	High	High	High

ACADEMY OF Management

Simple Statistical Embeddings

- *One-hot encoding*
- *Bag-of-Words*
- *Count Vectorization*
- *N-grams*
- *TF-IDF*

Neural network-based Embeddings

- **Word2Vec**
- **GloVe**
- **ELMo**
- **BERT**
- **GPT**

Basic Text Analyses

- *Information extraction*
- *Semantic Similarity*
- *Text Matching*
- *Sentiment Capturing*

Advanced Text Analyses

- *Information extraction*
- *Semantic Similarity*
- *Text Matching*
- *Sentiment Capturing*

…and more!

- *Coreference Resolution*
- *Part of Speech Tagging*
- *Question Answering*
- *Text Generation*

ACADEMY OF Management

TF-IDF

In [1]:
```
#Lib loading
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer

#Data loading
corpus = [
    'An embedding is a numerical map of a piece of information',
    'An embedding translates text into meaning',
    'morse code is a simple and somewhat primitive form of embedding'
]

#Analysis Excution
Tfid = TfidfVectorizer()
model = Tfid.fit_transform(corpus)

#Result Printout
df = pd.DataFrame(model.toarray(), columns=Tfid.get_feature_names_out())
print(df)
```

```
         an       and      code  embedding      form  information      into  \
0  0.271977  0.000000  0.000000   0.211214  0.000000     0.357617  0.000000
1  0.342620  0.000000  0.000000   0.266075  0.000000     0.000000  0.450504
2  0.000000  0.342884  0.342884   0.202513  0.342884     0.000000  0.000000

         is       map   meaning     morse  numerical        of     piece  \
0  0.271977  0.357617  0.000000  0.000000   0.357617  0.543954  0.357617
1  0.000000  0.000000  0.450504  0.000000   0.000000  0.000000  0.000000
2  0.260772  0.000000  0.000000  0.342884   0.000000  0.260772  0.000000

   primitive    simple  somewhat      text  translates
0   0.000000  0.000000  0.000000  0.000000    0.000000
1   0.000000  0.000000  0.000000  0.450504    0.450504
2   0.342884  0.342884  0.342884  0.000000    0.000000
```

ACADEMY OF Management

Word2Vec

In [1]:
```
#Lib loading
import gensim

#Data loading
corpus = [['An', 'embedding', 'is', 'a', 'numerical', 'map', 'of', 'a', 'piece', 'of', 'information'],
          ['An', 'embedding', 'translates', 'text', 'into', 'meaning']]

#Model Training
model_cbow = gensim.models.Word2Vec(corpus, min_count=1, vector_size=100, window=5, epochs=10)

#Result Printout
vector = model_cbow.wv['embedding']
print(vector)
```

```
[ 9.2157527e-05  3.0722953e-03 -6.8130689e-03 -1.3727958e-03
  7.6694302e-03  7.3449980e-03 -3.6763391e-03  2.6463254e-03
 -8.3177658e-03  6.2038335e-03 -4.6404880e-03 -3.1691839e-03
  9.3077878e-03  8.7519846e-04  7.4875532e-03 -6.0709226e-03
  5.1612263e-03  9.9217566e-03 -8.4525691e-03 -5.1402324e-03
 -7.0656864e-03 -4.8608002e-03 -3.7822053e-03 -8.5392678e-03
  7.9624271e-03 -4.8443913e-03  8.4169200e-03  5.2688336e-03
 -6.5523260e-03  3.9589470e-03  5.4729977e-03 -7.4248393e-03
 -7.4024280e-03 -2.4719061e-03 -8.6305914e-03 -1.5833091e-03
 -4.0489889e-04  3.3023157e-03  1.4467249e-03 -8.7958865e-04
 -5.5930102e-03  1.7329209e-03 -9.0232148e-04  6.7927754e-03
  3.9693485e-03  4.5285127e-03  1.4357996e-03 -2.6980657e-03
 -4.3655573e-03 -1.0267758e-03  1.4398774e-03 -2.6450581e-03
 -7.0720674e-03 -7.8014233e-03 -9.1183875e-03 -5.9379213e-03
 -1.8490404e-03 -4.3249056e-03 -6.4576617e-03 -3.7135798e-03
  4.2844685e-03 -3.7375225e-03  8.3812205e-03  1.5333984e-03
 -7.2459462e-03  9.4339745e-03  7.6333391e-03  5.4869382e-03
 -6.8484894e-03  5.8250097e-03  4.0099821e-03  5.1883170e-03
  4.2563854e-03  1.9330455e-03 -3.1697650e-03  8.3516622e-03
  9.6146045e-03  3.7870838e-03 -2.8338626e-03  5.9834583e-06
  1.2175719e-03 -8.4586581e-03 -8.2249688e-03 -2.2707264e-04
  1.2373109e-03 -5.7453122e-03 -4.7292765e-03 -7.3492411e-03
  8.3317403e-03  1.2519081e-04 -4.5109652e-03  5.7046567e-03
  9.1843726e-03 -4.0992382e-03  7.9671536e-03  5.3730868e-03
  5.8767116e-03  5.1810953e-04  8.2179587e-03 -7.0144339e-03]
```

ACADEMY OF Management

BERT

In [1]:
```
#Lib Loading
import pandas as pd
from transformers import TFBertModel, AutoTokenizer

model_checkpoint = 'bert-base-cased'
tokenizer = AutoTokenizer.from_pretrained(model_checkpoint)
fe_model = TFBertModel.from_pretrained(model_checkpoint)
fe_model.trainable = False  # Freeze the BERT model


#Data Loading
corpus = [
    'An embedding is a numerical map of a piece of information',
    'An embedding translates text into meaning',
    'morse code is a simple and somewhat primitive form of embedding'
]


#Model Building
input_ids = []
attention_masks = []
embeddings = []
cls_embedding=[]

for sent in corpus:
    encoded_dict = tokenizer(sent, is_split_into_words=False,
                             add_special_tokens = True,
                             max_length = 20,
                             padding='longest',
                             return_attention_mask = True,
                             truncation=True,
                             return_tensors = 'tf')

    input_ids.append(encoded_dict['input_ids'])
    attention_masks.append(encoded_dict['attention_mask'])
    outputs = fe_model(encoded_dict)
    sentence_embedding = outputs.last_hidden_state[:, 0, :].numpy().squeeze()
    cls_embedding.append(sentence_embedding)
    embeddings.append(outputs[0])
```

```
Some weights of the PyTorch model were not used when initializing the TF 2.0 model TFBertModel: ['cls.seq_relationship.bias',
'cls.predictions.transform.dense.bias', 'cls.seq_relationship.weight', 'cls.predictions.transform.dense.weight', 'cls.predictio
ns.transform.LayerNorm.bias', 'cls.predictions.bias', 'cls.predictions.transform.LayerNorm.weight']
- This IS expected if you are initializing TFBertModel from a PyTorch model trained on another task or with another architectur
e (e.g. initializing a TFBertForSequenceClassification model from a BertForPreTraining model).
- This IS NOT expected if you are initializing TFBertModel from a PyTorch model that you expect to be exactly identical (e.g. i
nitializing a TFBertForSequenceClassification model from a BertForSequenceClassification model).
All the weights of TFBertModel were initialized from the PyTorch model.
If your task is similar to the task the model of the checkpoint was trained on, you can already use TFBertModel for predictions
without further training.
```

```
fe_model.trainable = False
```

Fine-tuning
(over 110m parameters)

or

Feature-extraction?

ACADEMY OF Management

BERT

Word-output (768,)

```
In [2]: print(embeddings[0][0][0])
```

```
-4.64401215e-01  9.73541662e-02 -3.72936785e-01 -3.29438597e-01
 9.80468690e-01 -3.00930113e-01  3.25070828e-01  5.00261784e-01
 6.33766726e-02 -5.56632429e-02 -3.88656616e-01 -2.12899089e-01
 1.63678482e-01 -3.01123202e-01  6.27575338e-01 -3.57154533e-02
-5.88731542e-02 -3.34739000e-01 -2.55192190e-01  8.68763924e-02
-1.82235032e-01 -1.24598570e-01 -3.67570221e-01 -2.99356654e-02
 8.92873853e-02  7.69404173e-02 -2.46629566e-02 -8.40695441e-01
-5.04726231e-01  3.65028113e-01  3.37164313e-01  1.80504933e-01
 4.58939932e-04 -1.53909266e-01  1.29691869e-01 -4.08318043e-01
 1.46229282e-01  2.03622356e-01 -4.83105242e-01 -5.04101589e-02
-2.63429806e-02  2.81769067e-01  1.87975969e-02 -2.24474788e+00
 3.74117009e-02  1.85864225e-01 -3.13252747e-01 -1.25783145e-01
-2.79865563e-01  2.95625180e-02  5.77151701e-02  1.42921135e-01
 1.78511262e-01  3.91036943e-02  2.00167465e+00 -8.89857188e-02
-2.69806206e-01 -4.58274037e-01  2.54827172e-01  8.99088681e-02
 1.24059945e-01 -1.25680566e-01 -4.44792956e-01 -4.21742469e-01
 1.90288401e+00  2.77344640e-02  2.56330788e-01 -9.41678211e-02
-2.90123701e-01 -2.86785156e-01 -2.73343682e-01  1.37615532e-01
-1.84836626e-01 -1.65684238e-01  2.53415257e-01  1.83138281e-01], shape=(768,), dtype=float32)
```

Sentence-output (768,)

```
In [3]: print(outputs.last_hidden_state[:, 0, :][0])
```

```
-6.29421830e-01  4.77918983e-02 -3.47426049e-02 -2.03885183e-01
 1.07147253e+00 -3.45974475e-01  1.31274343e-01  3.18293482e-01
 3.26574668e-02  7.88378902e-03 -2.41035223e-01 -3.35012496e-01
 9.14501473e-02 -2.57445663e-01  4.09186095e-01  3.54113504e-02
 8.75370204e-02 -3.02496701e-01 -2.53423601e-01  2.11689427e-01
-3.84841383e-01 -4.80080582e-02 -1.60304636e-01 -1.80559129e-01
-5.27200103e-03 -7.68583566e-02 -1.35316551e-01 -7.02031434e-01
-3.03306013e-01  4.06021744e-01  2.67805755e-01  3.19206536e-01
-5.65475523e-02 -1.91613510e-01  2.55124032e-01 -1.69807017e-01
 1.09871641e-01  1.65860325e-01 -4.20414001e-01 -2.38324806e-01
-3.99055257e-02  3.72895449e-01 -4.35848758e-02 -2.61843586e+00
 9.52361748e-02  1.39490545e-01 -1.84165239e-01 -4.34509628e-02
-2.35709995e-01  2.01754004e-01  9.91596282e-02  2.22252548e-01
 9.49729085e-02 -1.82498191e-02  2.20636082e+00 -2.94232517e-01
-1.46470964e-01 -2.78755069e-01 -4.24390063e-02 -4.04349007e-02
 1.63667232e-01  5.52126579e-02 -4.91207570e-01 -2.83667058e-01
 1.60457695e+00  2.27619752e-01  3.12596470e-01 -3.81770656e-02
-3.77781421e-01 -3.61503772e-02 -2.75348544e-01  1.91947877e-01
-1.11367606e-01 -3.26290965e-01  2.75407404e-01  8.81668478e-02], shape=(768,), dtype=float32)
```

ACADEMY OF Management

- **Tie with the Theory (Task)**
- **Tie with the Data**
- **Search Similar Tasks**
- **Trial and Error**
- **Record and Replicate**

ACADEMY OF Management

Simple embedding techniques:

Reid, S. W., McKenny, A. F., & Short, J. C. (2023). Synthesizing best practices for conducting dictionary-based computerized text analysis research. In *Methods to Improve Our Field* (Vol. 14, pp. 43-78). Emerald Publishing Limited.

Harrison, J. S., Josefy, M. A., Kalm, M., & Krause, R. (2023). Using supervised machine learning to scale human-coded data: A method and dataset in the board leadership context. *Strategic Management Journal*, *44*(7), 1780-1802.

Machine learning-based language model:

Harrison, J. S., Thurgood, G. R., Boivie, S., & Pfarrer, M. D. (2019). Measuring CEO personality: Developing, validating, and testing a linguistic tool. *Strategic Management Journal*, *40*(8), 1316-1330.

Guo, W., Sengul, M., & Yu, T. (2021). The impact of executive verbal communication on the convergence of investors' opinions. *Academy of Management Journal*, *64*(6), 1763-1792.

Transformer-based language model:

Miric, M., Jia, N., & Huang, K. G. (2023). Using supervised machine learning for large-scale classification in management research: The case for identifying artificial intelligence patents. *Strategic Management Journal*, *44*(2), 491-519.

Carlson, N. A. (2023). Differentiation in microenterprises. *Strategic Management Journal*, *44*(5), 1141-1167.

ACADEMY OF Management

Embeddings

Thank you!

Questions?

rsong@lsu.edu