NLP

Natural Language Processing

Overview

- Motivation
- NLP overview
- Text Classification
- Sentiment Analysis
  - BERT
  - finBERT

Motivation

- Other than time series data, most financial data is unstructured, much of it in text format, and language specific
  - Financial reports, 10Ks
  - Earnings calls
  - News stories
  - Analyst reports
  - Excel spreadsheets?
  - Arguably, even financial numerical data is not well structured for machine consumption (e.g. tables)

What's difficult about human language?

- There are many languages, with different rules.
- Unstructured data
- Multiple meanings
  - Bank account
  - River bank
  - Sick burn
- Context
- Written language has errors
- Sparsity – language has many words, but a piece of text can have very few in comparison

What is Natural Language Processing?

- Natural Language Processing broadly refers to the **study and development of computer systems that can interpret speech and text as humans naturally speak and type it.**
- Human communication is frustratingly **vague** at times; we all use colloquialisms, abbreviations, and don't often bother to correct misspellings. These inconsistencies **make computer analysis of natural language difficult at best.**
- But in the last decade, both NLP techniques and machine learning algorithms have progressed immeasurably.

Three aspects to any chunk of text

- **Semantic Information** - the specific **meaning** of an individual word
  - “the bat flew through the air” can have multiple meanings depending on the definition of bat: winged mammal, wooden stick, or something else entirely? Knowing the relevant definition is vital for understanding the meaning of a sentence.
- **Syntax Information** - sentence or phrase **structure**
  - “Sarah joined the group already with some search experience.” Who exactly has the search experience here? Sarah, or the group? Depending on how you read it, the sentence has very different meaning with respect to Sarah’s abilities.
- **Context Information** - context that a word, phrase, or sentence appears in
  - What is the **concept being discussed**? If a person says that something is “sick”, are they talking about healthcare or video games?

Text component example

- Let's analyze the sentence, "**Billy hit the ball over the house**." Taken separately, the three types of information would return:
  - Semantic information: person – act of striking an object with another object – spherical play item – place people live
  - Syntax information: subject – action – direct object – indirect object
  - Context information: this sentence is about a child playing with a ball
- These aren't very helpful by themselves. They indicate a vague idea of what the sentence is about, but **full understanding requires the successful combination of all three components**.
- This analysis can be accomplished in a number of ways, through machine learning models or by inputting rules for a computer to follow when analyzing text, each with drawbacks:
  - Rules based – takes a lot of time, hard to generalize
  - ML – not a lot of labeled data

NLP & ML

- Natural Language Processing (NLP) deals with how computers understand and translate human language.
- With NLP, machines can make sense of written or spoken text and perform tasks like translation, keyword extraction, topic classification, and more.
- But to automate these processes and deliver accurate responses, you'll need machine learning (plus some rules).
  - Very early text mining systems were entirely based on rules and patterns.

NLP Use Cases

- Language Translation
- Sentiment Analyses - Identifies emotions in text and classifies opinions as positive, negative, or neutral.
- Text Extraction - enables you to pull out pre-defined information from text.
  - Helps you recognize and extract relevant keywords and features (like product codes, colors, and specs), and named entities (like names of people, locations, company names, emails, etc).
- Topics Classification - Helps you organize unstructured text into categories.
- Chatbots - AI systems designed to interact with humans through text or speech.

Credit: Rob Dobi

IDEAS MADE TO MATTER | FINANCE

Why finance is deploying natural language processing

by Tracy Mayor | Nov 3, 2020

Why It Matters

As the amount of textual data increases, natural language processing is becoming a strategic tool for financial analysis. Here are three ways it helps.

Share

NLP can improve decision-making and speed inside financial organizations:

- **Automation.** NLP can replace the manual processes financial institutions employ to turn unstructured data into a more usable form — for example, automating capture of earnings calls, management presentations, and acquisition announcements.

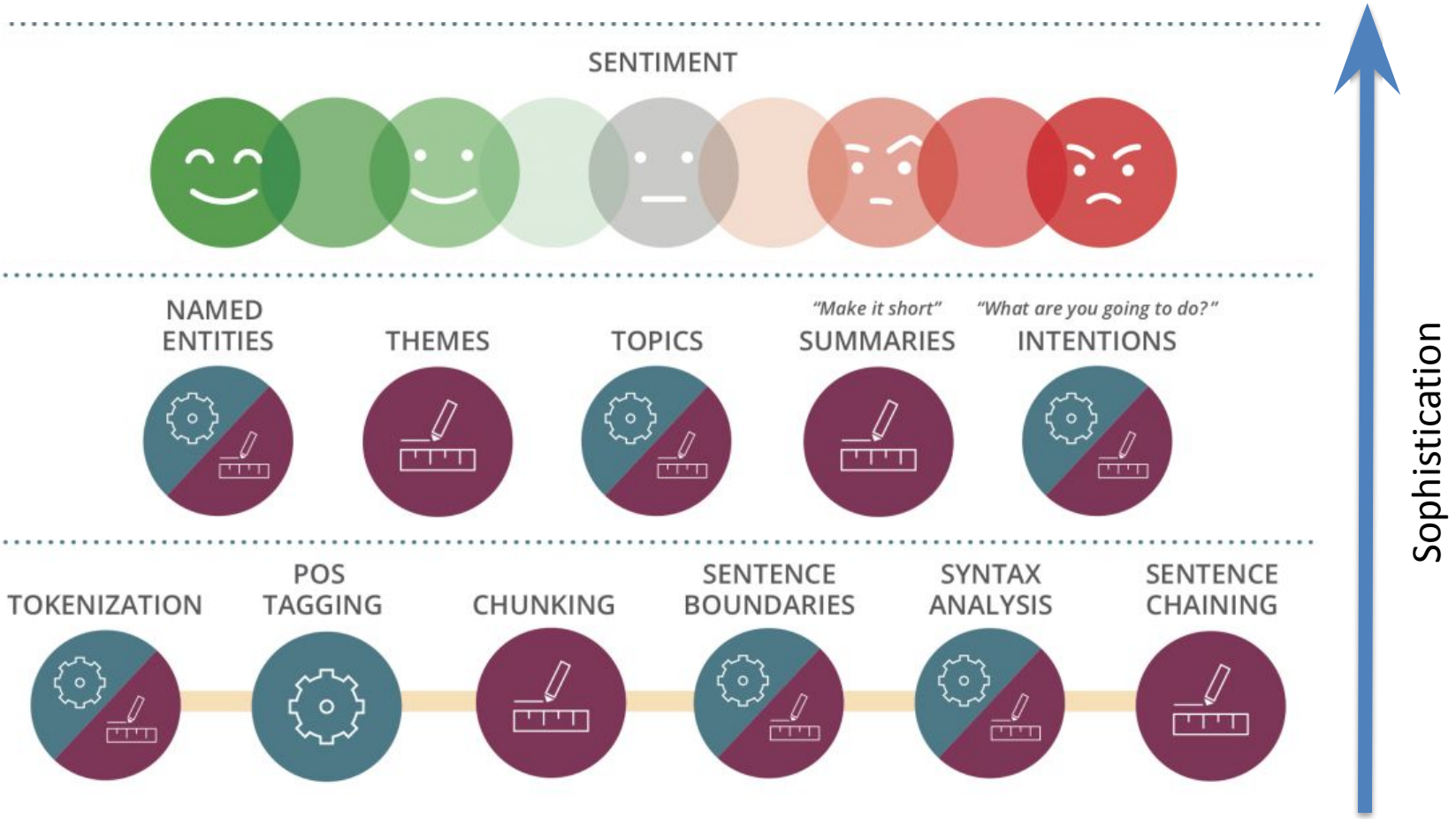
- **Data enrichment.** Once unstructured data is captured, adding context makes it more searchable and actionable. "Imagine I get a transcript of that earnings call, and I want to find places where they're talking about environmental impact," Shulman said. Machine learning can enrich that raw text with metadata — flagging sections that address environmental impact, financial impact, or other topics of interest.

- **Search and discovery.** Finance is on a mission to find competitive advantage in broader and more varied types of data, but what's missing is a search experience that's as effortless and effective as the Google search bar consumers are accustomed to.

NLP in finance use cases

- Risk assessments
  - NLP techniques use multiple data points to assess credit risk.
  - Mistakes in loan data extraction can lead to the wrong assessments. Named entity recognition (NER), an NLP technique, is useful in such situations.
- Financial sentiment
  - see if/how the market will react to news and whether the stock price will fall or rise.
  - FinBERT is one of the models developed for the financial services sector.
- Accounting and auditing
  - identify, focus, and visualize anomalies in the day-to-day transactions.
  - aid with the identification of significant potential risks and possible fraud, like money laundering.
- Portfolio selection and optimization
  - NLP can be utilized for semi-log-optimal portfolio optimization. Semi-log-optimal portfolio selection is a computational alternative to the log-optimal portfolio selection.
- Stock behavior predictions
  - Deep learning combined with NLP outmatches previous methodologies working with financial time series to a great extent.

NLP Tasks


SENTIMENT
NAMED ENTITIES
THEMES
TOPICS
"Make it short"
SUMMARIES
"What are you going to do?"
INTENTIONS
TOKENIZATION
POS TAGGING
CHUNKING
SENTENCE BOUNDARIES
SYNTAX ANALYSIS
SENTENCE CHAINING
Sophistication

NLP Techniques

- **Syntactic analysis** – or parsing – analyzes text using basic grammar rules to identify sentence **structure**, how words are organized, and how words relate to each other.
- **Semantic analysis** focuses on capturing the **meaning** of text.
  - First, it studies the meaning of each individual word (lexical semantics).
  - Then, it looks at the combination of words and what they mean in context.

Syntactic Analysis Tasks

- **Tokenization** consists of breaking up a text into smaller parts called tokens (which can be sentences or words) to make text easier to handle.
- **Part of speech tagging (PoS tagging)** labels tokens as verb, adverb, adjective, noun, etc. This helps infer the meaning of a word (for example, the word “book” means different things if used as a verb or a noun).
- **Lemmatization & stemming** consist of reducing inflected words to their base form to make them easier to analyze.
- **Stop-word removal** removes frequently occurring words that don’t add any semantic value, such as I, they, have, like, yours, etc.

Stemming & Lemmatization

- **Stemming** usually refers to a crude heuristic process that **chops off the ends of words** in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes.
- **Lemmatization** usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to **return the base or dictionary form of a word**, which is known as the lemma.

Stemming & Lemmatization

Form	Suffix	Stem
studies	-es	studi
studying	-ing	study
niñas	-as	niñ
niñez	-ez	niñ

Form	Morphological information	Lemma
studies	Third person, singular number, present tense of the verb study	study
studying	Gerund of the verb study	study
niñas	Feminine gender, plural number of the noun niño	niño
niñez	Singular number of the noun niñez	niñez

Semantic Analyses Tasks

- Word sense disambiguation tries to identify in which sense a word is being used in a given context.
- Relationship extraction attempts to understand how entities (places, persons, organizations, etc) relate to each other in a text.

Sparsity Challenge

- A set of numbers (e.g. vector, matrix, etc.), is considered **sparse** when a high percentage of the values are assigned a constant default value
- Text data requires a special approach to machine learning. This is because text data can have hundreds of thousands of dimensions (words and phrases) but tends to be very sparse.
- For example, the English language has around 100,000 words in common use. But any given tweet only contains a few dozen of them.

Tokenization

- Tokenization involves breaking a text document into pieces that a machine can understand, such as words.
- English is especially easy. See all this white space between the letters and paragraphs? That makes it really easy to tokenize. So, NLP rules are sufficient for English tokenization.
- But how do you teach a machine learning algorithm what a word looks like? And what if you're not working with English-language documents? Logographic languages like Mandarin Chinese have no whitespace.
- This is where we use machine learning for tokenization.

PoS Tagging

- Part of Speech Tagging (PoS tagging) means identifying each token's part of speech (noun, adverb, adjective, etc.) and then tagging it as such.
- PoS tagging forms the basis of a number of important Natural Language Processing tasks.
- We need to correctly identify Parts of Speech in order to recognize entities, extract themes, and to process sentiment.

Named Entity Recognition

- At their simplest, named entities are people, places, and things (products) mentioned in a text document.
- Unfortunately, entities can also be hashtags, emails, mailing addresses, phone numbers, and Twitter handles.
- In fact, just about anything can be an entity if you look at it the right way.
- It’s also important to note that Named Entity Recognition models rely on accurate PoS tagging from those models.

Sentiment Analysis

- Sentiment analysis is the process of determining whether a piece of writing is positive, negative or neutral, and then assigning a weighted sentiment score to each entity, theme, topic, and category within the document.
- This is an incredibly complex task that varies wildly with context. For example, take the phrase, “sick burn” In the context of video games, this might actually be a positive statement.
- Creating a set of NLP rules to account for every possible sentiment score for every possible word in every possible context would be impossible.
- But by training a machine learning model on pre-scored data, it can learn to understand what “sick burn” means in the context of video gaming, versus in the context of healthcare.
- **Unsurprisingly, each language requires its own sentiment classification model.**

Categorization and Classification

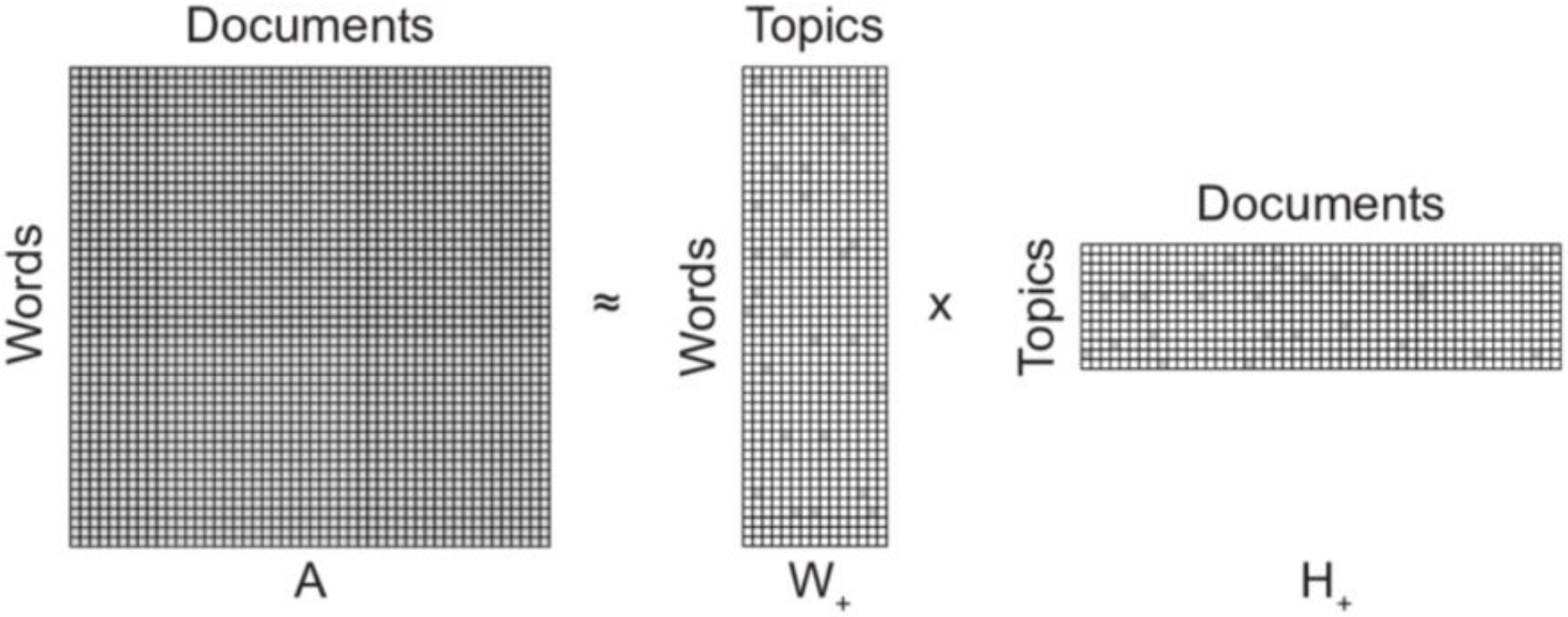
- Categorization means sorting content into buckets to get a quick, high-level overview of what's in the data.
- To train a text classification model, data scientists use pre-sorted content and gently shepherd their model until it's reached the desired level of accuracy.
- The result is accurate, reliable categorization of text documents that takes far less time and energy than human analysis.

Unsupervised ML for NLP and Text Analytics

- **Clustering** means grouping similar documents together into groups or sets. These clusters are then sorted based on importance and relevancy (hierarchical clustering).
- Another type of unsupervised learning is **Latent Semantic Indexing (LSI)**. This technique identifies on words and phrases that frequently occur with each other. Data scientists use LSI for faceted searches, or for returning search results that aren't the exact search term.
- For example, the terms "manifold" and "exhaust" are closely related documents that discuss internal combustion engines. So, when you Google "manifold" you get results that also contain "exhaust".
- **Matrix Factorization** is another technique for unsupervised NLP machine learning. This uses "latent factors" to break a large matrix down into the combination of two smaller matrices. Latent factors are similarities between the items.
- **Unsupervised learning is tricky, but far less labor- and data-intensive than its supervised counterpart.**

Matrix Factorization

$$A \approx W \times H$$



Further Reading:
https://www.analyticsvidhya.com/blog/2021/06/part-15-step-by-step-guide-to-master-nlp-topic-modelling-using-nmf/

(some) Python Libraries for NLP

- Natural Language Toolkit(NLTK)
- GenSim
- SpaCy
- CoreNLP
- TextBlob
- AllenNLP
- polyglot
- scikit-learn

TEXT CLASSIFICATION

What is Text Classification?

- Text classification is a machine learning technique that assigns a set of predefined **categories** to open-ended text. Text classifiers can be used to organize, structure, and categorize pretty much any kind of text – from documents, medical studies and files, and all over the web.
- For example, new articles can be organized by topics; support tickets can be organized by urgency; chat conversations can be organized by language; brand mentions can be organized by sentiment; and so on.
- Text classification is one of the fundamental tasks in natural language processing with broad applications such as sentiment analysis, topic labeling, spam detection, language and intent detection.

Example

- Here's an example of how it works:
  - "The user interface is quite straightforward and easy to use."
- A text classifier can take this phrase as an input, analyze its content, and then automatically assign relevant **tags**, such as UI and Easy To Use.
- Why are tags important?
  - A "classification"
  - Better search algos, proprietary context tags specific to your business, e.g. finance, retail, etc.

Why is Text Classification Important?

- It's estimated that around **80% of all information is unstructured**, with text being one of the most common types of unstructured data.
  - Because of the messy nature of text, analyzing, understanding, organizing, and sorting through text data is hard and time-consuming, so **most companies fail to use it to its full potential**.
- Using text classifiers, companies can **automatically structure all manner of relevant text**, from emails, legal documents, social media, chatbots, surveys, and more in a fast and cost-effective way.
- This allows companies to **save time** analyzing text data, **automate business processes**, and **make data-driven business decisions.**

Why use ML for Text Classification?

- Scalability
  - Manually analyzing and organizing is slow and much less accurate. Machine learning can automatically analyze millions of surveys, comments, emails, etc., at a fraction of the cost, often in just a few minutes. Text classification tools are scalable to any business needs, large or small.
- Real-time analysis
  - There are critical situations that companies need to identify as soon as possible and take immediate action (e.g., PR crises on social media). Machine learning text classification can follow your brand mentions constantly and in real time, so you'll identify critical information and be able to take action right away.
- Consistent criteria
  - Human annotators make mistakes when classifying text data due to distractions, fatigue, and boredom, and human subjectivity creates inconsistent criteria. Machine learning, on the other hand, applies the same lens and criteria to all data and results. Once a text classification model is properly trained it performs with unsurpassed accuracy.

How is it done?

- You can perform text classification in two ways: **manual** or **automatic**.
- Manual text classification involves a **human annotator**, who interprets the content of text and **categorizes** it accordingly. This method can deliver good results but it's time-consuming and expensive.
- Automatic text classification applies machine learning, natural language processing (NLP), and other AI-guided techniques to automatically classify text in a faster, more cost-effective, and more accurate manner.

Automatic Text Classification

There are many approaches to automatic text classification, but they all fall under three types of systems:

- Rule-based systems
  - Rule-based approaches classify text into organized groups by using a set of handcrafted linguistic rules. These rules instruct the system to use semantically relevant elements of a text to identify relevant categories based on its content. Each rule consists of an antecedent or pattern and a predicted category.
  - Pros
    - Human comprehensible and can be improved over time
  - Cons
    - Require deep knowledge, time-consuming, difficult to maintain
- Machine learning-based systems
- Hybrid systems

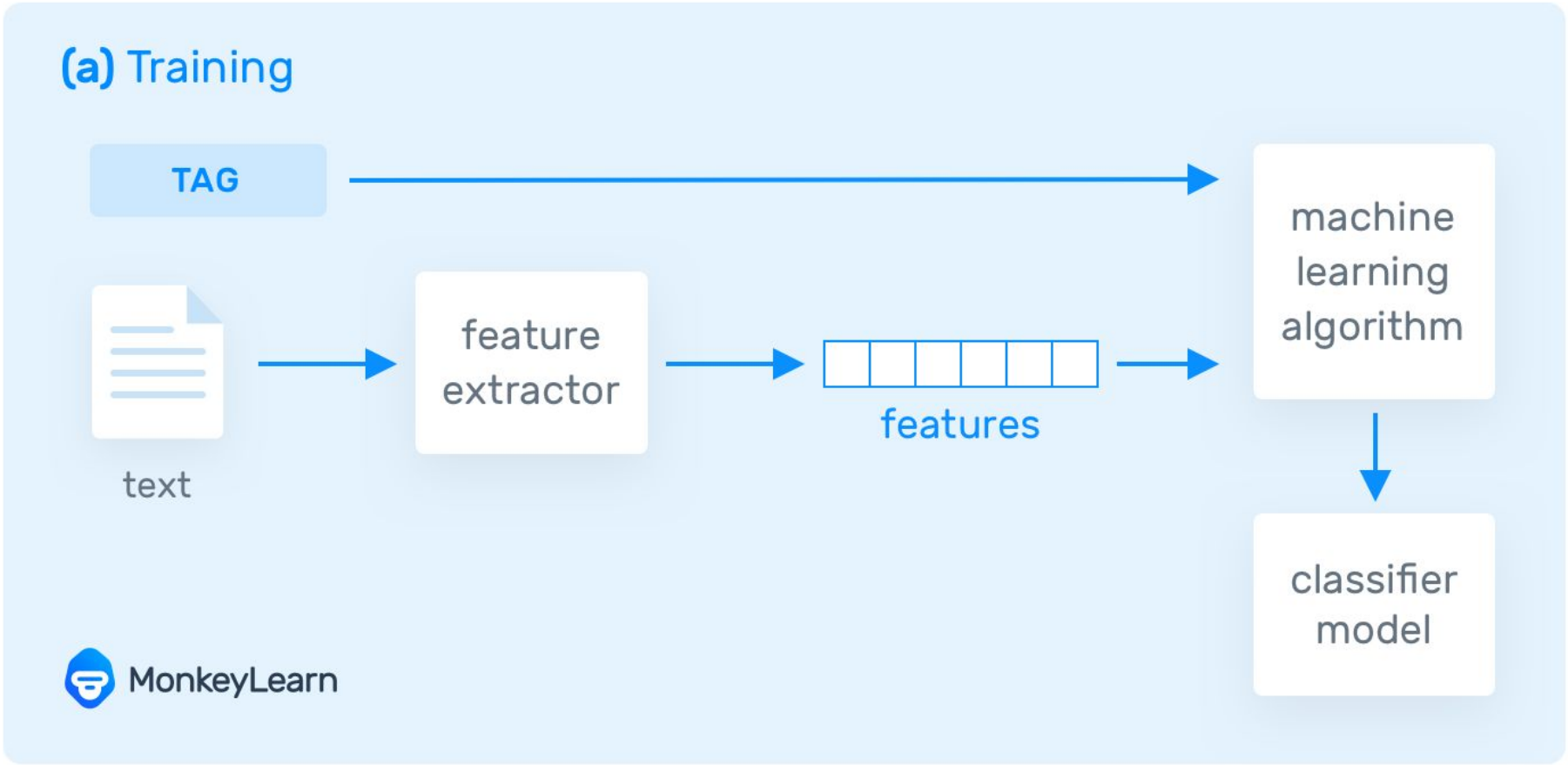
ML TC

- Instead of relying on manually crafted rules, machine learning text classification learns to make classifications based on past observations.
- By using pre-labeled examples as training data, machine learning algorithms can learn the different associations between pieces of text, and that a particular output (i.e., tags) is expected for a particular input (i.e., text).
- A “tag” is the pre-determined classification or category that any given text could fall into.

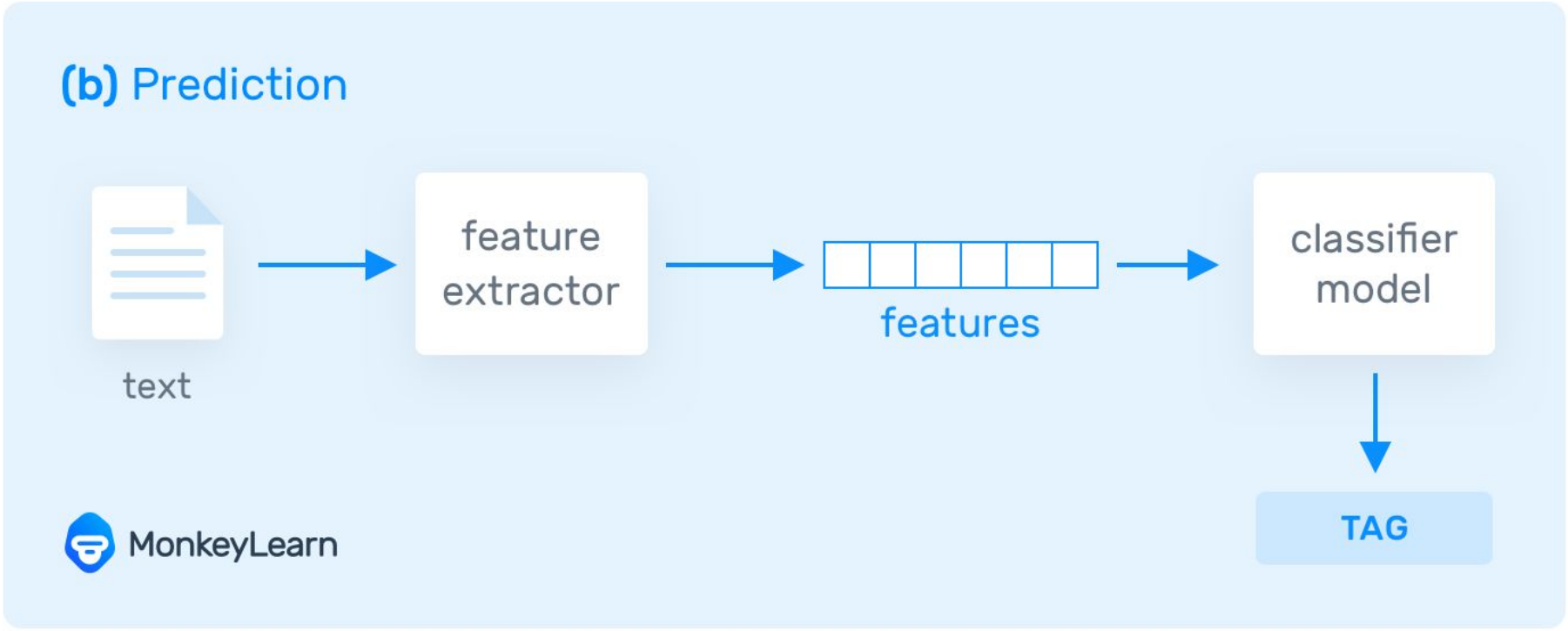
Text Feature Extraction

- The first step towards training a machine learning NLP classifier is feature extraction: a method is used to transform each text into a numerical representation in the form of a vector.
- One of the most frequently used approaches is **bag of words (BoW)**, where a vector represents the frequency of a word in a predefined dictionary of words.
  - https://machinelearningmastery.com/gentle-introduction-bag-words-model/
- For example, if we have defined our dictionary to have the following words **{This, is, the, not, awesome, bad, basketball}**, and we wanted to vectorize the text "This is awesome," we would have the following vector representation of that text: **(1, 1, 0, 0, 1, 0, 0)**.
- Then, the machine learning algorithm is fed with training data that consists of pairs of feature sets (vectors for each text example) and tags (e.g. sports, politics) to produce a classification model.
- Alternative to BoW: Word Embeddings

ML TC Training



ML TC Prediction



Popular ML algos for TC

- Naive Bayes
  - The Naive Bayes family of statistical algorithms are some of the most used algorithms in text classification and text analysis, overall.
  - One of the members of that family is Multinomial Naive Bayes (MNB) with a huge advantage, that you can get really good results even when your dataset isn't very large (~ a couple of thousand tagged samples) and computational resources are scarce.
- Support Vector Machines
- More info here.
- Deep Learning
  - Deep learning algorithms, like Word2Vec or GloVe are also used in order to obtain better vector representations for words and improve the accuracy of classifiers trained with traditional machine learning algorithms.

Further Info

- APIs – many text classifier APIs are available
  - MonkeyLearn
    - https://monkeylearn.com/api/v3/#classify
  - AWS
  - Google
    - https://cloud.google.com/natural-language#section-2
  - Others
- Demo
  - https://monkeylearn.com/text-classification/#examples

FINBERT SENTIMENT ANALYSES

NLP lack of data challenge

- One of the biggest challenges in NLP is the shortage of training data.
- Most task-specific datasets contain only a few thousand or a few hundred thousand ***human-labeled*** training examples.
- Researchers have developed a variety of techniques for training general purpose language representation models using the enormous amount of unannotated text on the web (known as pre-training).
- **The pre-trained model** can then be fine-tuned on small-data NLP tasks like question answering and sentiment analysis, resulting in substantial accuracy improvements compared to training on these datasets from scratch.

BERT

- **Bidirectional Encoder Representations from Transformers**, or BERT.
- Anyone can train their own state-of-the-art question answering system (or a variety of other models).
- What Makes BERT Different?
  - BERT is the first deeply bidirectional, unsupervised language representation, pre-trained using only a plain text corpus (in this case, Wikipedia)
  - Breakthrough research on **transformers**: models that process words in relation to all the other words in a sentence, rather than one-by-one in order.
  - BERT models consider the full context of a word by looking at the words that come before and after it—particularly useful for understanding intended meaning.
- Paper: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding by Devlin Mchang Lee Toutanova, Google AI Language
  - https://arxiv.org/pdf/1810.04805.pdf (2019)

BERT: Why does this matter?

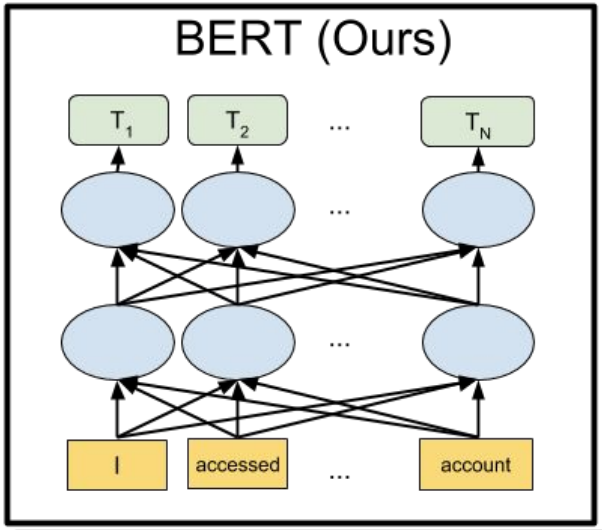
- Pre-trained representations can either be **context-free** or **contextual**, and contextual representations can further be **unidirectional** or **bidirectional**.
- Context-free models such as **word2vec** or **GloVe** generate a single word embedding representation for each word in the vocabulary.
  - For example, the word “bank” would have the same context-free representation in “bank account” and “bank of the river.”
- Contextual models instead generate a representation of each word that is based on the other words in the sentence.
  - For example, in the sentence “I accessed the bank account,” a unidirectional contextual model would represent “bank” based on “I accessed the” but not “account.”
- BERT represents “bank” using both its previous and next context — “I accessed the ... account” — starting from the very bottom of a deep neural network, making it deeply bidirectional.

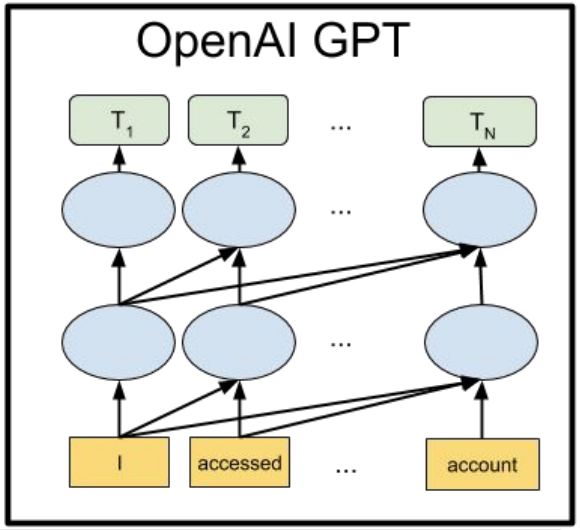
BERT Innovations

- It borrowed the transformer (T of BERT) architecture from machine translation, which does a better job of modeling long-term dependencies than RNN-based ones (excellent overview here).
- It introduced the Masked Language Modeling (MLM) task, where a random 15% of all tokens are masked and the model predicts them, enabling true bi-directionality (B of BERT).
- Intuitive explanations of how transformers and BERT can be found here and here.

Contextual NN Architectures

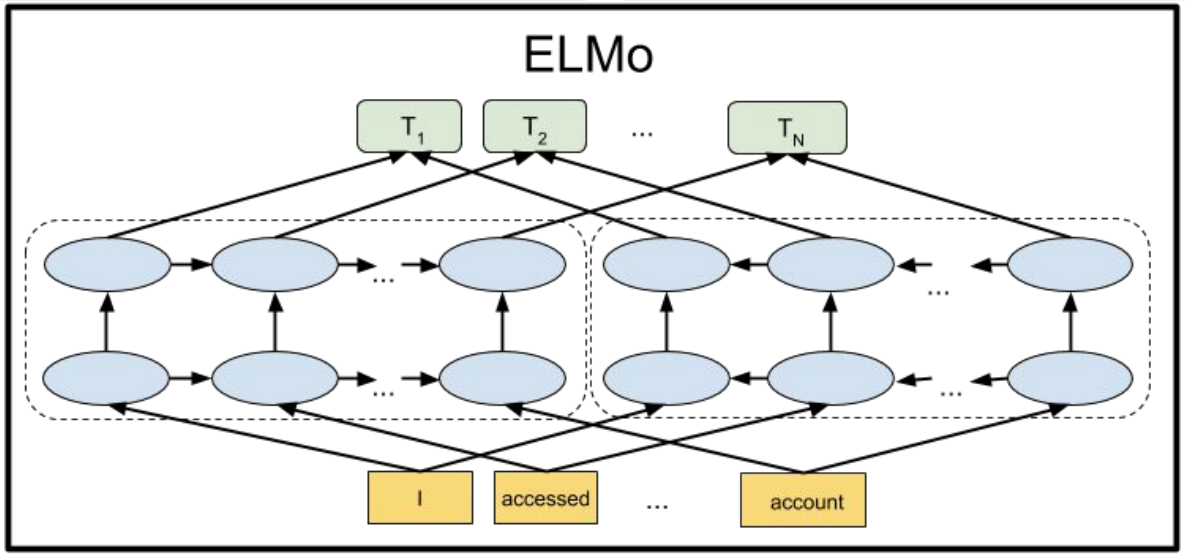
Deeply Bidirectional





Unidirectional

Shallowly Bidirectional



Details: https://jalammar.github.io/illustrated-bert/

BERT Performance Results from 2018

SQuAD1.1 Leaderboard

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar et al. '16)	82.304	91.221
1 Oct 05, 2018	BERT (ensemble) *Google AI Language* https://arxiv.org/abs/1810.04805	**87.433**	**93.160**
2 Sep 09, 2018	nlnet (ensemble) *Microsoft Research Asia*	85.356	91.202
3 Jul 11, 2018	QANet (ensemble) *Google Brain & CMU*	84.454	90.490

https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html

finBERT

- FinBERT is a pre-trained NLP model based on BERT.
- Put simply: FinBERT is just a version of BERT trained on financial data (hence the "Fin" part), specifically for sentiment analysis.
- Remember: BERT is a general language model. Financial news and stock reports often involve a lot of domain-specific jargon, so a model like BERT isn't really able to generalize well in this domain.
- You'd see similar problems using BERT on, say, legal filings or medical literature.

How to use finBert

- The data used to train FinBERT is text from financial news services, as well as the FiQA dataset.
- The total corpora size is 4.9B tokens.
  - Corporate Reports 10-K & 10-Q: 2.5B tokens
  - Earnings Call Transcripts: 1.3B tokens
  - Analyst Reports: 1.1B tokens
- You can use FinBERT in two ways:
  - Pre-trained model. You can fine-tune FinBERT with your own dataset. FinBERT is most suitable for financial NLP tasks.
  - Fine-tuned model. If you are interested in simply using FinBERT for financial sentiment classification prediction, a fine-tuned FinBERT model that is fine tuned on 10,000 manually annotated analyst statements is provided.

finBERT Case Study

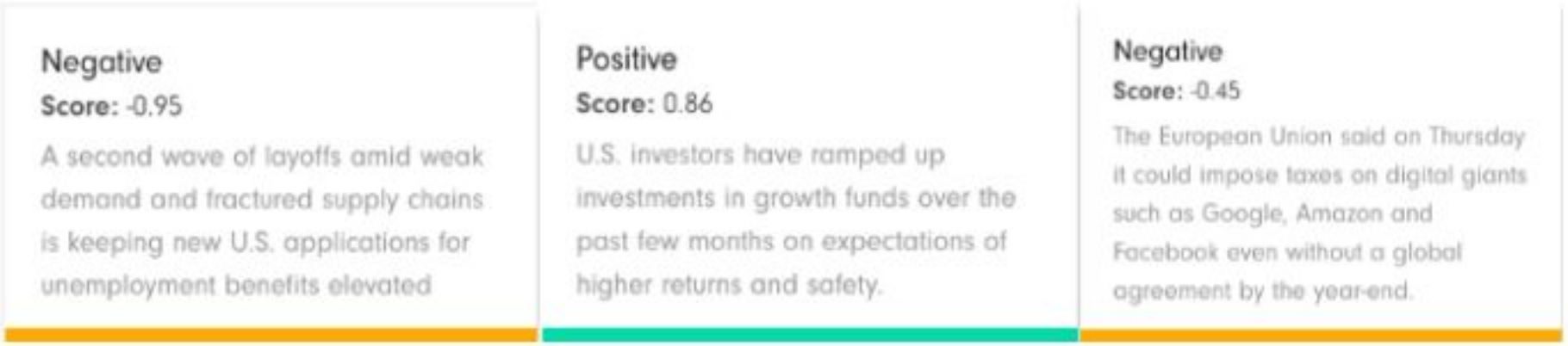
- Financial Sentiment Analysis
  - Used the dataset of Financial Phrasebank.
  - It is a very well thought-out and carefully labeled albeit a small dataset.
  - Researchers extracted 4500 sentences from various news articles, which include financial terms.
  - Then 16 experts and master students with finance backgrounds labeled them.
  - They didn't only report labels but also inter-annotator agreement level for each sentence, which means how many experts labeled as positive, neutral and negative.
- Source: https://medium.com/prosus-ai-tech-blog/finbert-financial-sentiment-analysis-with-bert-b277a3607101

finBERT Results

	All data			Data with 100% agreement		
Model	Loss	Accuracy	F1 Score	Loss	Accuracy	F1 Score
1. LSTM	0.81	0.71	0.64	0.57	0.81	0.74
2. LSTM with ELMo	0.72	0.75	0.7	0.50	0.84	0.77
3. ULMFit	0.41	0.83	0.79	0.20	0.93	0.91
4. LPS	-	0.71	0.71	-	0.79	0.80
5. HSC	-	0.71	0.76	-	0.83	0.86
6. FinSSLX	-	-	-	-	0.91	0.88
FinBERT	**0.37**	**0.86**	**0.84**	**0.13**	**0.97**	**0.95**

Experimental results on the Financial PhraseBank dataset

Prediction Example



Some prediction examples from FinBERT

Where does FinBERT fail?

- FinBERT is very good on phrases annotators agreed on
  - 97% accuracy on PhraseBank subset where annotators agreed 100%
- Annotator agreements (disagreements)
  - Positive-negative 98.7% agreement
  - Negative-neutral 94.2% agreement
  - Positive-neutral 75.2% agreement
    - difficulty in separating “commonly used company glitter with actual positive statements”
  - Paper: https://arxiv.org/abs/1307.5336

Where does FinBERT fail?

- finBert misclassifications
  - **73% are between positive and neutral labels**
  - 5% are between positive and negative labels
- Examples

> *A. Pre-tax loss totaled euro 0.3 million, compared to a loss of euro 2.2 million in the first quarter of 2005 .*
>
> **Label: Positive, Predicted: Negative**

> *B. This implementation is very important to the operator, since it is about to launch its Fixed to Mobile convergence service in Brazil*
>
> **Label: Neutral, Predicted: Positive**

> *C. The situation of coated magazine printing paper will continue to be weak*
>
> **Label: Negative, Predicted: Neutral**

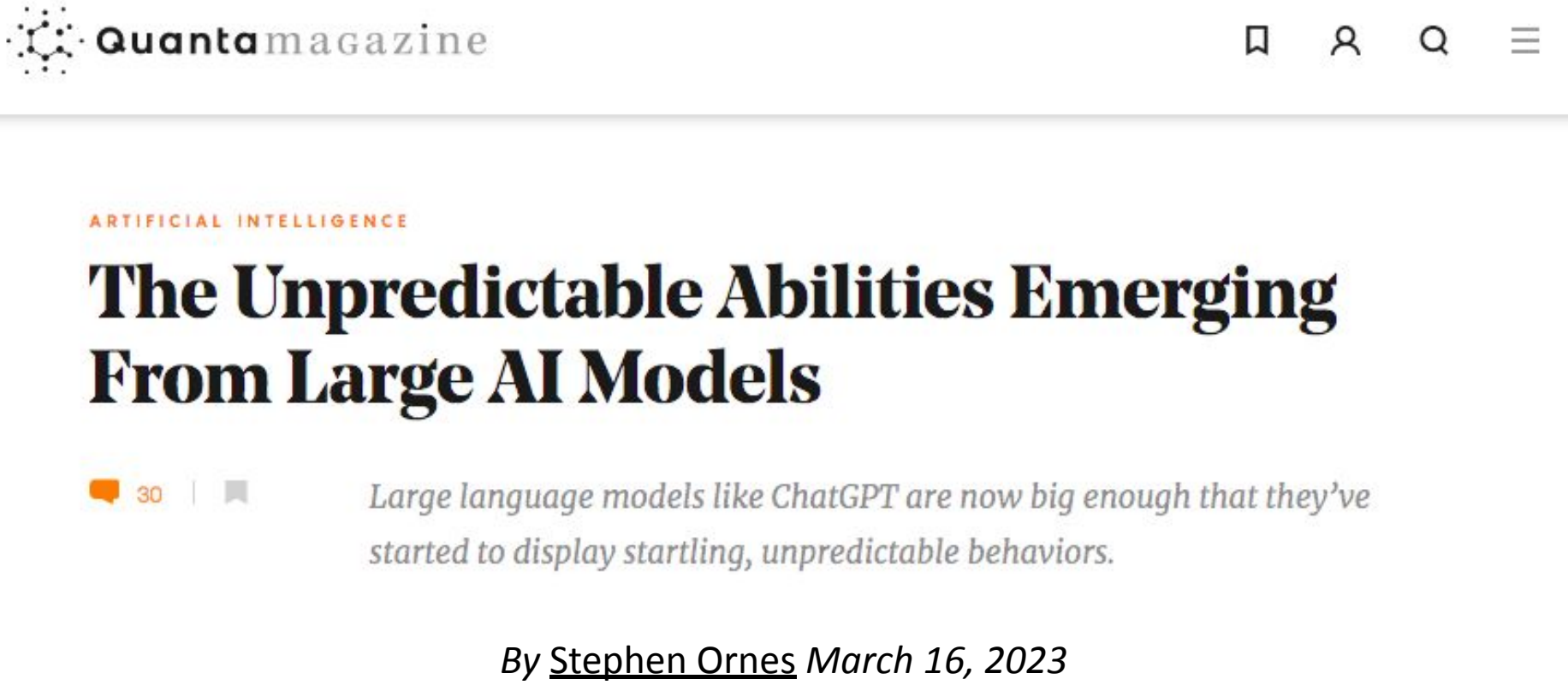
LLMs

LARGE LANGUAGE MODELS

What movie do these emojis describe?

- Simple LLM: “The movie is a movie about a man who is a man who is a man”
- Medium LLM: “The Emoji Movie”
- Complex LLM: “Finding Nemo”
- One of 204 tasks to test LLM performance

LLM “Technological Step Function”?



Source: https://www.quantamagazine.org/the-unpredictable-abilities-emerging-from-large-ai-models-20230316/

LLMs and Surprising Results

- OpenAI GPT-3 - 175 billion parameters
- Google's PaLM - can be scaled up to 540 billion
- Users began describing more and more emergent behaviors.
  - One engineer reported being able to convince ChatGPT that it was a Linux terminal and getting it to run some simple mathematical code to compute the first 10 prime numbers. It could finish the task faster than the same code running on a real Linux machine.

So what's going on here?

- "Despite trying to expect surprises, I'm surprised at the things these models can do"
  - It's surprising because these models supposedly have one directive: to accept a string of text as input and predict what comes next, over and over, based purely on statistics.
- Computer scientists anticipated that scaling up would boost performance on **known tasks**, but they didn't expect the models to suddenly handle so many **new, unpredictable ones**.

Zero-Shot Learning

- As with the movie emoji task, researchers had no reason to think that a language model built to predict text would convincingly imitate a computer terminal.
- Many of these emergent behaviors illustrate “zero-shot” or “few-shot” learning, which describes an LLM’s ability to solve problems it has never — or rarely — seen before.
- This has been a long-time goal in artificial intelligence research.

Emergence

- Biologists, physicists, ecologists and other scientists use the term "**emergent**" to describe self-organizing, collective behaviors that appear when a large collection of things acts as one.
  - Combinations of lifeless atoms give rise to living cells; water molecules create waves;
  - Murmurations of starlings swoop through the sky in changing but identifiable patterns;
  - Cells make muscles move and hearts beat.
- Critically, emergent abilities show up in systems that involve lots of individual parts. But researchers have only recently been able to document these abilities in LLMs as those models have grown to enormous sizes.

Emergent Abilities (EAs)

- Tasks that big models can complete that smaller models can't, many of which seem to have little to do with analyzing text.
- They range from multiplication to generating executable computer code to, apparently, decoding movies based on emojis.
- New analyses suggest that for some tasks and some models, there's a **threshold of complexity** beyond which the functionality of the model skyrockets.
  - They also suggest a dark flip side: As they increase in complexity, some models reveal new biases and inaccuracies in their responses.
- List of EAs so far

Emergent Abilities = Overfitting?

- Are they using training data to generate these answers?
- Is the training data representative of the "true" distribution from where these questions and answers are coming from?
- "reach beyond the constraints of their training data"
- No, not overfitting on training data.

LM History

- Language models have been around for decades.
- Until about five years ago, the most powerful were based on what's called a **recurrent neural network.**
  - These essentially take a string of text and predict what the next word will be.
  - What makes a model "recurrent" is that it learns from its own output: Its predictions feed back into the network to improve future performance.

Transformers

- In 2017, researchers at Google Brain introduced a new kind of architecture called a **transformer**.
- While a recurrent network analyzes a sentence word by word, the transformer **processes all the words at the same time**.
- This means transformers can process big bodies of text in **parallel**.

Beyond Imitation

- Possible reasons for emergence:
  - As suggested by comparisons to biological systems, larger models truly do gain new abilities spontaneously.
    - “It may very well be that the model has learned something fundamentally new and different that it didn’t have at a smaller size,”
  - May be the culmination of an internal, statistics-driven process that works through chain-of-thought-type reasoning.
    - Large LLMs may simply be learning heuristics that are out of reach for those with fewer parameters or lower-quality data.
  - Something else

“Notorious Liars”

- LLMs are convincing liars
  - Not all the time, but enough to be wary
  - Need to check and verify
- Emergence leads to unpredictability, and unpredictability — which seems to increase with scaling — makes it difficult for researchers to **anticipate the consequences of widespread use**.
- As models improve their performance when scaling up, they **may also increase the likelihood of unpredictable phenomena, including those that could potentially lead to bias or harm.**
  - Can tell the bot to not be biased and self-correct