Less supervision?

—

L101: Machine Learning for Language Processing

Andreas Vlachos

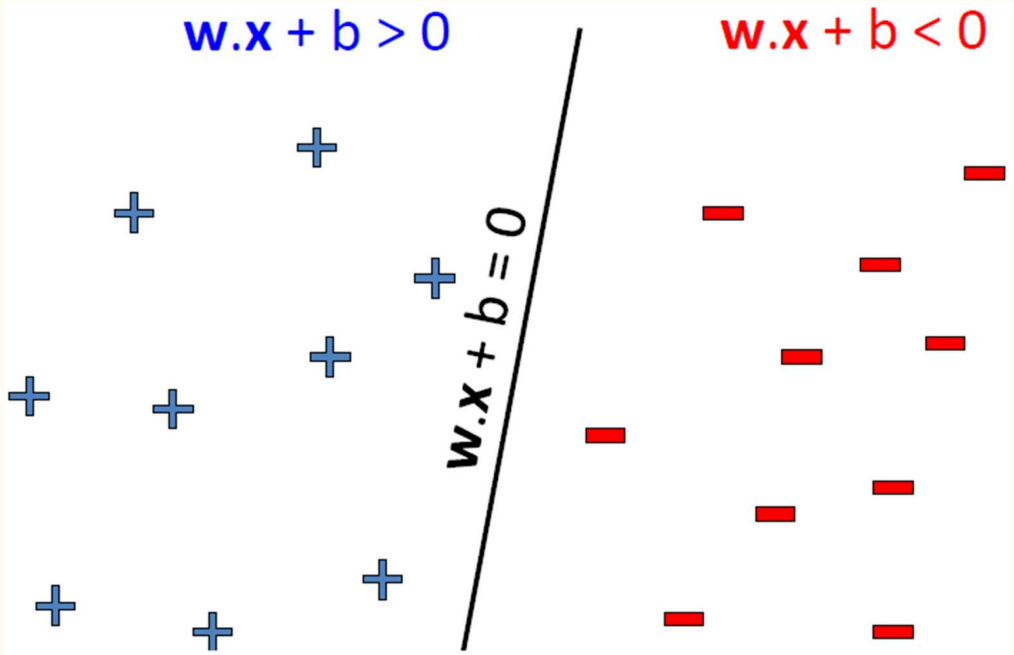
So far: supervised learning

Why?

- Training models using real outputs for the task you want to solve has better chances of success
- Even if we didn't need labeled data for training, we still want to know how good our model is before deploying it (and not tune hyperparams on the test data!)

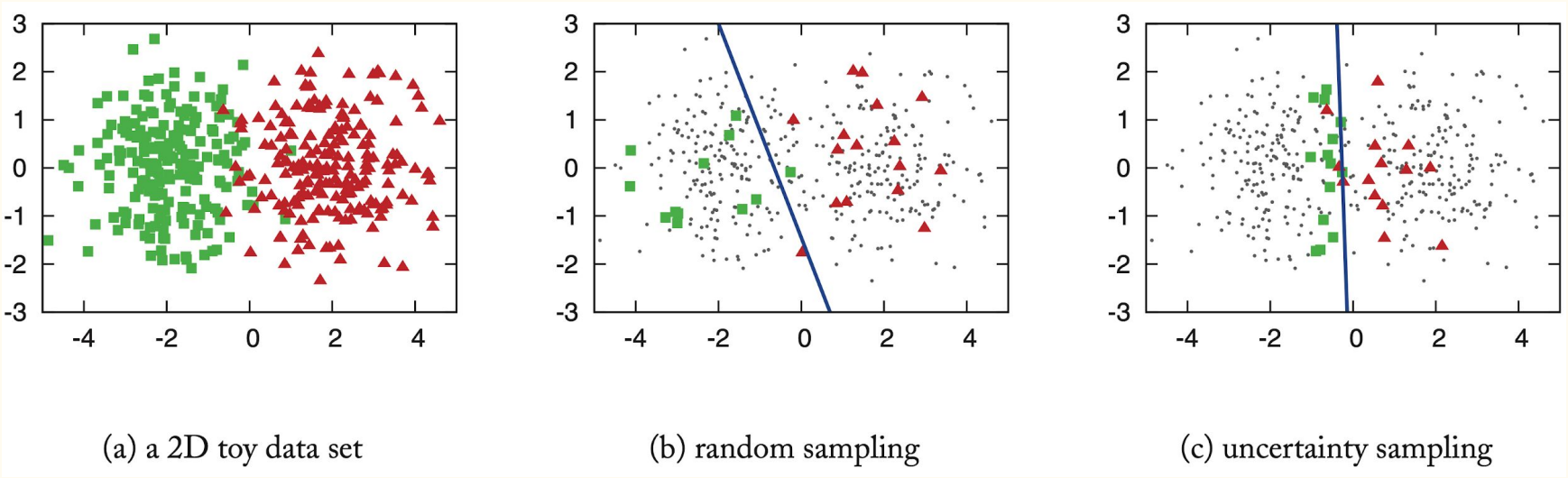
But it needs labeled data for training, how can we help ourselves given that we can only find unlabeled data for free (usually!)

Do we need all that labeled data for training?



- Are all instances equally useful in learning a model?
- Let the algorithm decide!
  - Typically select a few instances to label, update model, repeat
- Active learning in education refers to the student asking questions

Active learning works



(a) a 2D toy data set (b) random sampling (c) uncertainty sampling

- Savings against randomly selected training data can be impressive
  - Even if you are fine-tuning BERT
- In fact less can be better (sometimes)
- But this is not always the case (more later)

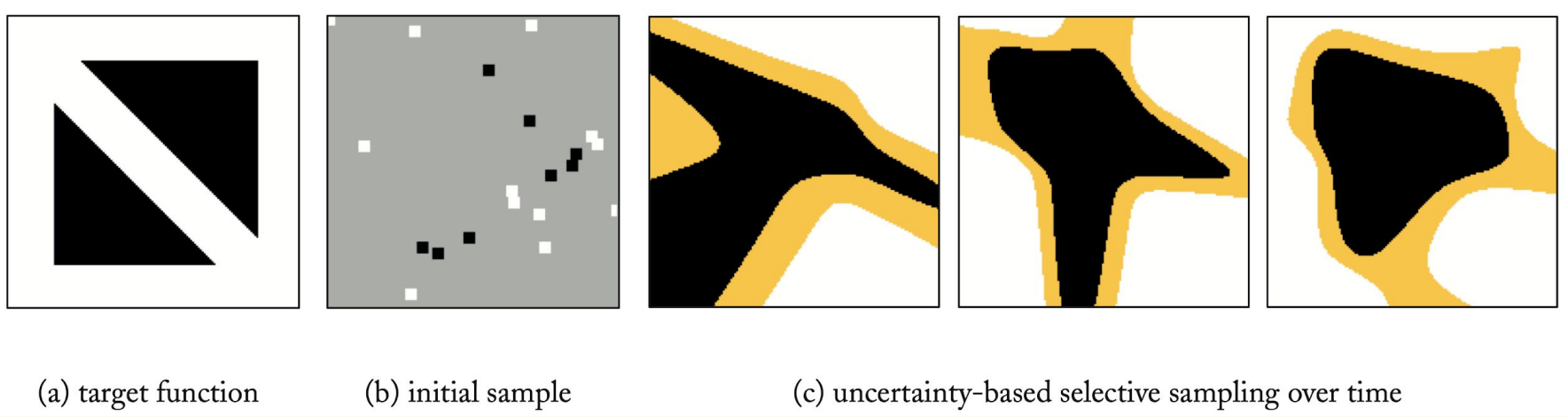
Active learning setups

- Pool-based
  - we have a fixed pool of unlabeled data and iterate through it
  - most common: we get some unlabelled instances and need to build a model
- Streaming/online:
  - decide whether to ask for a label now or never
  - ask a customer if the transaction is fraudulent, if the e-mail is spam, etc.
- Constructing instances:
  - generate instances for labeling
  - very rare in NLP; need very good generative models
  - now seems within reach
- Feature-based active learning
  - label features instead of instances

How to choose informative instances?

- Uncertainty based sampling
  - Least confident: pick the instances with the lowest score by the model for any label
  - Entropy in the label distribution (for probabilistic models)
- Query by committee: train a few models and select the instances where they disagree the most
- Meta-learning: learn a model to select the most useful instances (pool, stream)
- Discriminate in favour of instances that don't look like the labeled data
- Bayesian Active Learning by Disagreement: combine uncertainty with informativity on model parameters
  - Selecting instances in batches is a common practical consideration

Things can go wrong



(a) target function (b) initial sample (c) uncertainty-based selective sampling over time

What would you do to avoid this?

- Make sure your initial sample is representative?
- Select instances at random too?

When not to active learn

- If we need data for evaluation
  - it should be obvious that AL biases the data, not guaranteed to be representative of the task
  - need to approach the data selection differently (active testing)
- If we don't think the model(s) we have can give good estimates of uncertainty
- If we are planning to change models later
  - Data selected by one model can be worse than random for another

Bandit learning

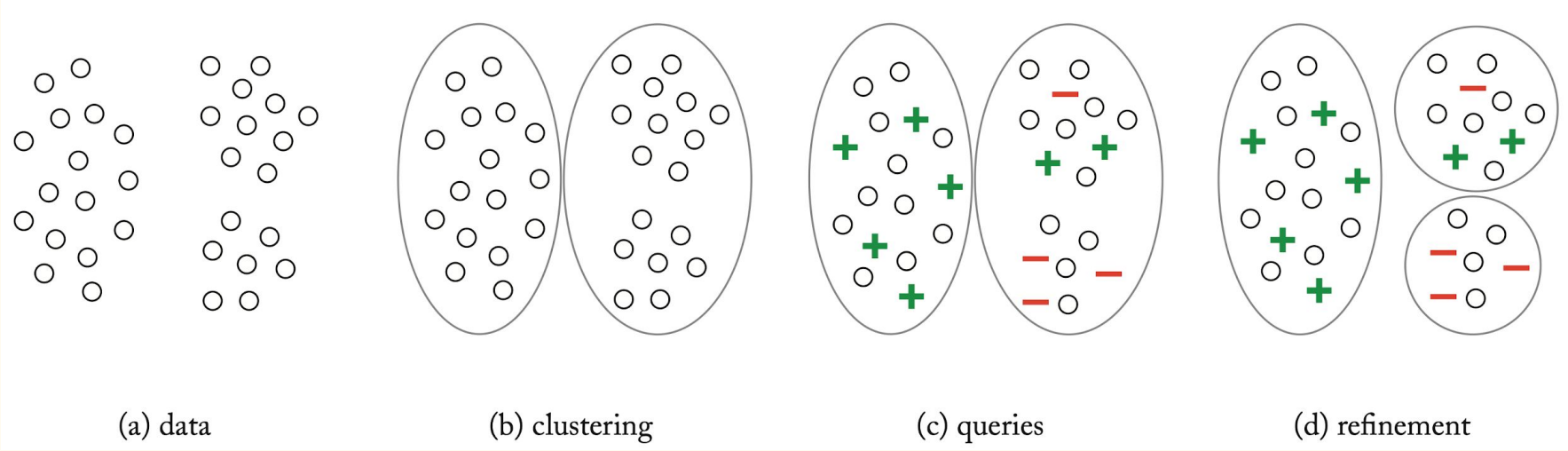
Some times obtaining a complete label for an instance is impossible: e.g. for a search query we need to check all webpages

Repeat:

- pick the most promising handle (= label)
- get a reward for that handle
- Update using the reward

The key here is how to define promising: a balance of exploration/exploitation
Can be adaptive: first explore, then exploit
See Kreutzer et al. (2017) for an application to NMT

Can we use the unlabeled data to help?



Yes, with clustering!

Hierarchical sampling: prefer instances from clusters that are impure

Clustering

Many good algorithms:

- K-means (++)
- Gaussian mixture models
- Spectral clustering
- Topic models can be thought of as soft clustering

They are great to explore unlabeled data and learn about theirs properties, but

Hard to evaluate...

Clustering: Science or Art: “the major obstacle is the difficulty in evaluating a clustering algorithm without taking into account the context”

Unsupervised NLP evaluation

Unsupervised [redacted] tagger output

1	2	3	4	1	5
There	are	70	children	there	.

What is the task?

CCG gold standard

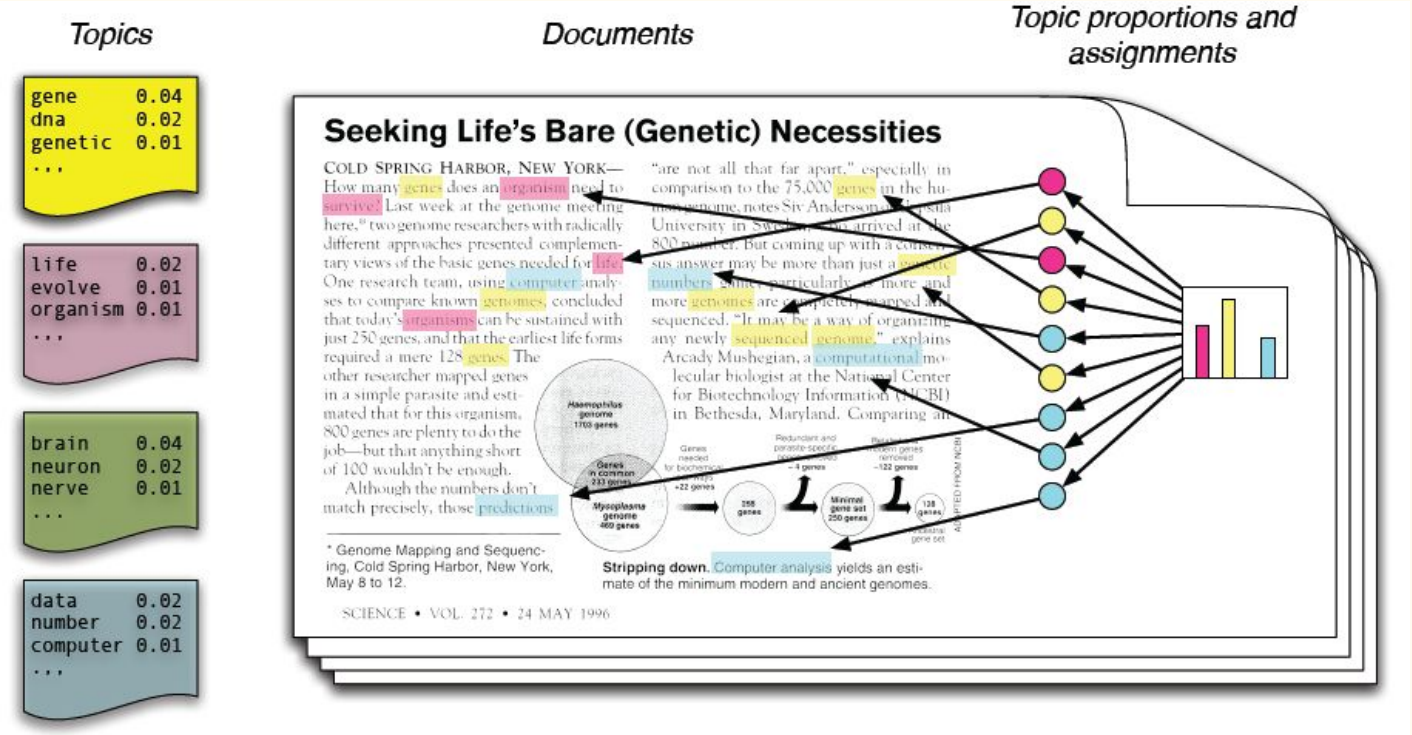
NP	(S\NP)/NP	N/N	N	(S\NP)\(S\NP)	.
There	are	70	children	there	.

Why PoS tagging and not CCG super-tagging?

So no point in doing unsupervised learning?

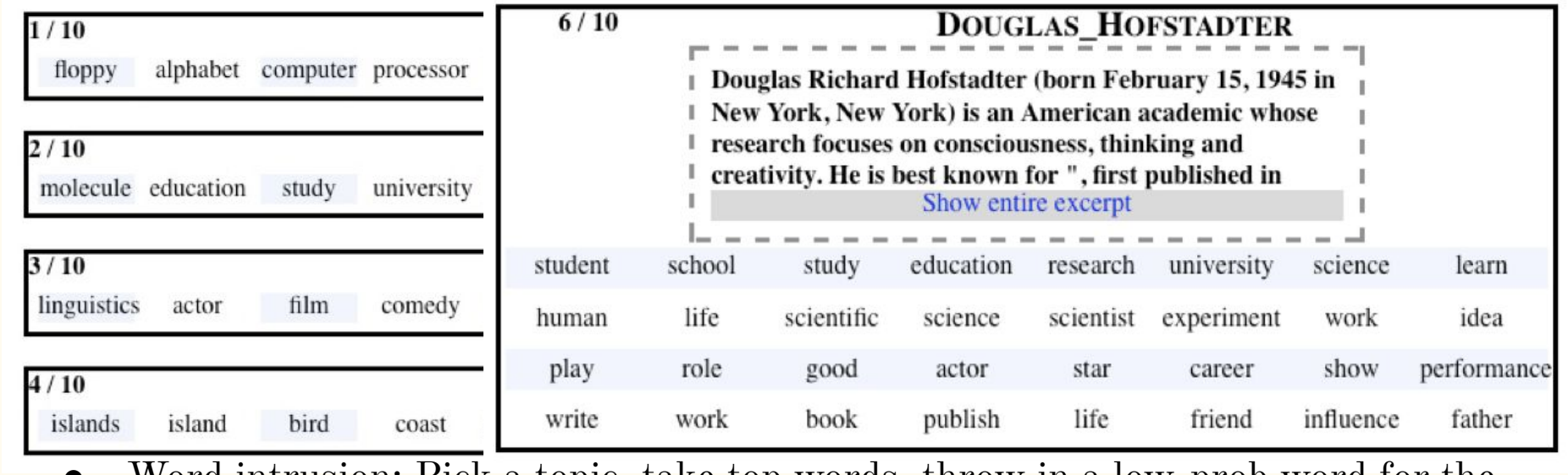
No, but you need to put it in **context** to evaluate it

Topic models



Blei (2011)

Topic modeling evaluation (reading tea leaves)



- Word intrusion: Pick a topic, take top words, throw in a low-prob word for the topic, ask humans to spot the intruder
- Topic intrusion: Pick a document, take its top topics, throw in a low-prob topic for the document, ask humans to spot the intruder
- Avoid automatic evaluation if you can

But isn’t unsupervised learning common?

Yes! But often this is not what unsupervised in the ML sense (no labeled data, e.g. in clustering no cluster info, in topic models no topic info):

- Some supervision gets in through dictionaries, mapping to labels, etc.
- Or (distant) supervision was readily available on the web
- Some dev set was most likely used (hopefully not the test set!)
- Nothing wrong with this; in fact it is a great way to solve tasks
- But if we want labels as output we need to provide them to the model

What about language modeling?

- Supervised or unsupervised learning?
- For me it is supervised, but we can harvest data for it at will
- More data beats better model

- The main application of LMs for a while was to score outputs from MT, ASR, etc.
- Nowadays?

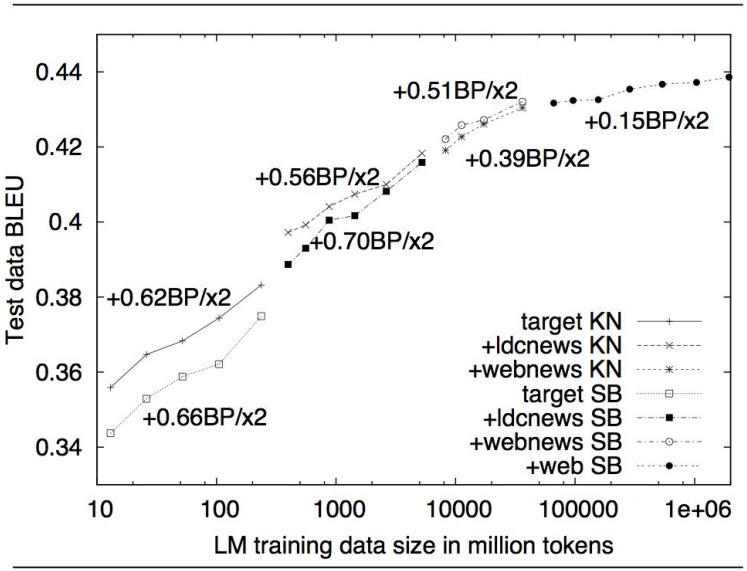


Figure 5: BLEU scores for varying amounts of data using Kneser-Ney (KN) and Stupid Backoff (SB).

Brants et al. (2007)

Word Embeddings

- Once a by-product of learning RNN-based language models, now a goal in itself
- That's the key insight of the word2vec paper: stop worrying about trying to build a language model, focus on the embeddings
- Supervised or unsupervised?
- Unsupervised: LM has a supervised training objective, but we don't have gold standard embeddings
  - This is also why their evaluation is difficult
  - Often done in context: input for supervised models (e.g. BERT)

Language models as models for task X?

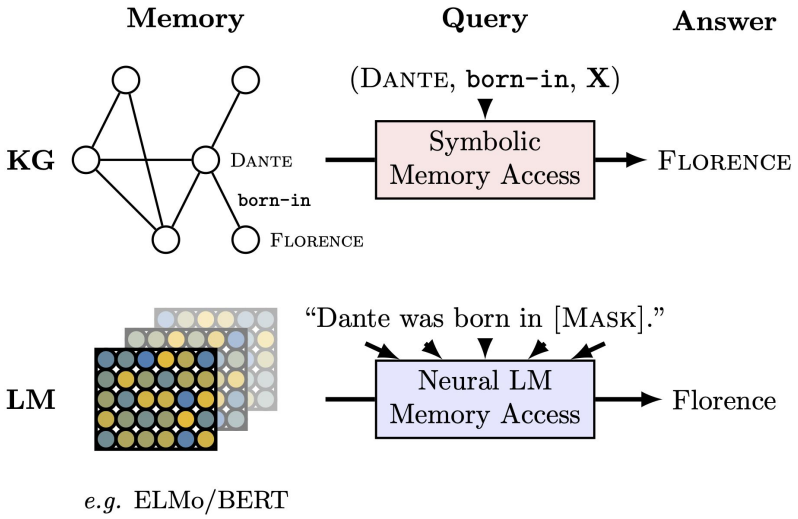


Figure 1: Querying knowledge bases (KB) and language models (LM) for factual knowledge.

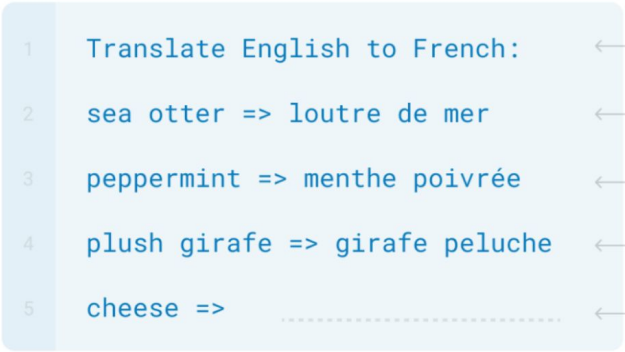
- Yes! But usually we need to add a bit task-specific supervision (fine-tuning)
- If our task can be modelled as an LM, we can take advantage of a lot of data and pre-trained models
- If we are building task-specific models, we'd better improve on it!
- If we developing a task/dataset, make sure a LM can't solve it (easily), e.g. don't use an LM (alone) to construct it

Petroni et al. (2019)

Prompting large language models

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



Brown et al. (2020)

- Look ma, no training, just a few examples!
- From feature engineering, to architecture, and now to prompt engineering?
- Not so fast:
  - Needs a lot of dev data for model/ prompt selection (Perez et al., 2021)
  - Fine-tuning is more reliable and not necessarily more expensive (Logan IV et al., 2021)
- Descriptions/instructions in zero-shot approaches can also be seen as prompts (Aly et al. 2021, Wei et al. 2022)

Bibliography

Active learning book (Burr Settles, where a lot of images where taken from)

Bandit learning book (Slivkins)

Foundations of LLMs (Xiao and Zhu, 2025)