Introduction to Large Language Models

Albert Gatt & Ayoub Bagheri

Utrecht University

Goals of this short course

- Main goal: understanding the fundamentals of (large) language models

→The language modeling objective

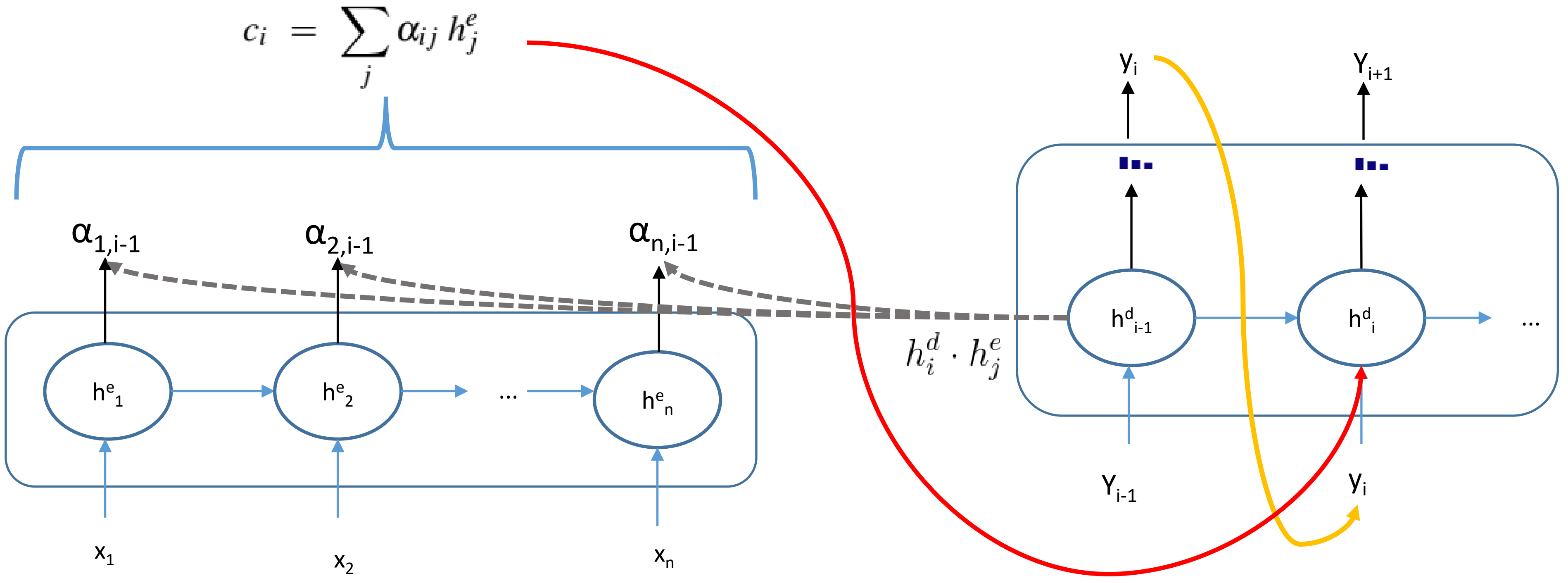
→Sequences and the encoder-decoder architecture

→Attention

→Transformer models

How the story ended last time

- Hopefully, during training, we achieve a way to identify, at each time-step in the decoder, which part of the source encoding is most relevant.



Generalising the idea of attention

- Transformer models are based on the following key ideas:
    1. We can take the notion of attention and generalise it. Fundamentally, it's a mechanism to learn meaningful relationships between elements. E.g.:
        - **Better meaning representations**: given a word, we can allow an encoder to learn its linguistic behaviour by looking at surrounding words.
        - **More fluent text generation**: given some context (a "prefix"), a decoder can learn to model the probability of next tokens by attending to all elements of the context.

    2. We can drop the idea of recurrence. (I.e. We no longer model sequences with RNNs)
        - This ovecomes memory problems (but incurs some costs).
        - It also means we can compute relationships between multiple elements in parallel (because we're not restricted to sequential processing).

Self-attention

The core operation in Transformers

Let’s generalize our terminology first

In the original E-D model with attention, we are computing a context vector which is a distribution over the encoder hidden states.

$$y_i = \sum_{j \leq i} \alpha_{ij} x_j$$

Sum the inputs seen so far, weighted by their relevance (alpha value)

$$\alpha_{ij} = \text{softmax}\left(score(x_i, x_j)\right) \forall j \leq i$$

Turn scores into weights, reflecting the relevance of each element j to element i

$$\text{score}(x_i, x_j) = x_i \cdot x_j$$

Score by “relatedness” or “similarity”

Inputs in self-attention computation play 3 different roles at different points.

- **Query**: the current focus element (i)
- **Key**: the other elements (j) to which we compare the query.

$$y_i = \sum_{j \le i} \alpha_{ij} x_j$$

- **Value**: used to compute the output for the current focus of attention

Self-Attention in Transformers: Single output

Basic idea: capture the key/query/value roles of the input through **learnable weight matrices**.

$$q_i = W^Q x_i; W^Q \in \mathbb{R}^{d \times d}$$

$$k_i = W^K x_i; W^K \in \mathbb{R}^{d \times d}$$

$$v_i = W^V x_i; W^V \in \mathbb{R}^{d \times d}$$

Here is what happens for a single output:

$$y_i = \sum_{j \le i} \alpha_{ij} \mathbf{v}_j$$

Final calculation is based on a weighted sum over the value vectors.

$$\alpha_{ij} = \text{softmax}\left(score(x_i, x_j)\right) \forall j \le i$$

Softmax (as before)

$$\text{score}(x_i, x_j) = \frac{\mathbf{q}_i \cdot \mathbf{k}_j}{\sqrt{d_k}}$$

Score is the dot product between **the query vector of x_i** and **other element's key vectors k_j**. All of dimensionality 1 * d.
Since this can become quite big, we scale it by a factor proportional to the vector dimension.

Self-Attention in Transformers: Entire sequence

Basic idea: capture the key/query/value roles of the input through **learnable weight matrices.**

$$q_i = W^Q x_i; W^Q \in \mathbb{R}^{d \times d}$$

$$k_i = W^K x_i; W^K \in \mathbb{R}^{d \times d}$$

$$v_i = W^V x_i; W^V \in \mathbb{R}^{d \times d}$$

Suppose input is of length N. Break it up into N tokens, and represent it as matrix X: one token per row. Column dimensions are **embeddings** (dense representations for the input tokens).

$$\mathbf{X} \in \mathbb{R}^{N \times d}$$

We can now precompute all the queries, keys and values…

$$\mathbf{X} \in \mathbb{R}^{N \times d}$$

$$\begin{aligned}\mathbf{Q} &= \mathbf{X}\mathbf{W}^Q \\ \mathbf{K} &= \mathbf{X}\mathbf{W}^K \\ \mathbf{V} &= \mathbf{X}\mathbf{W}^V\end{aligned}$$

Also, we can:

- compute all the Q/K comparisons
- Scale
- Apply softmax
- Multiply by V

$$\mathrm{SelfAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

What about order?

Simple solution:

- Combine the input representations X with **positional embeddings**.
  - E.g. *x = Janet will back the bill*.

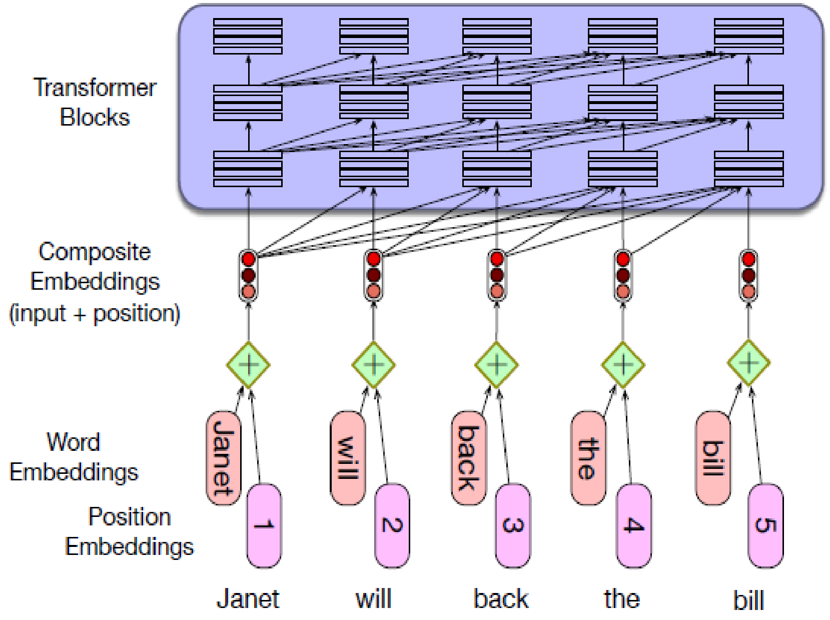


Image from: Jurafsky & Martin, Ch. 9

Problem:

For the edge cases (e.g. words at the end of very long sentences) we will have very few examples. These embeddings will be poorly trained.

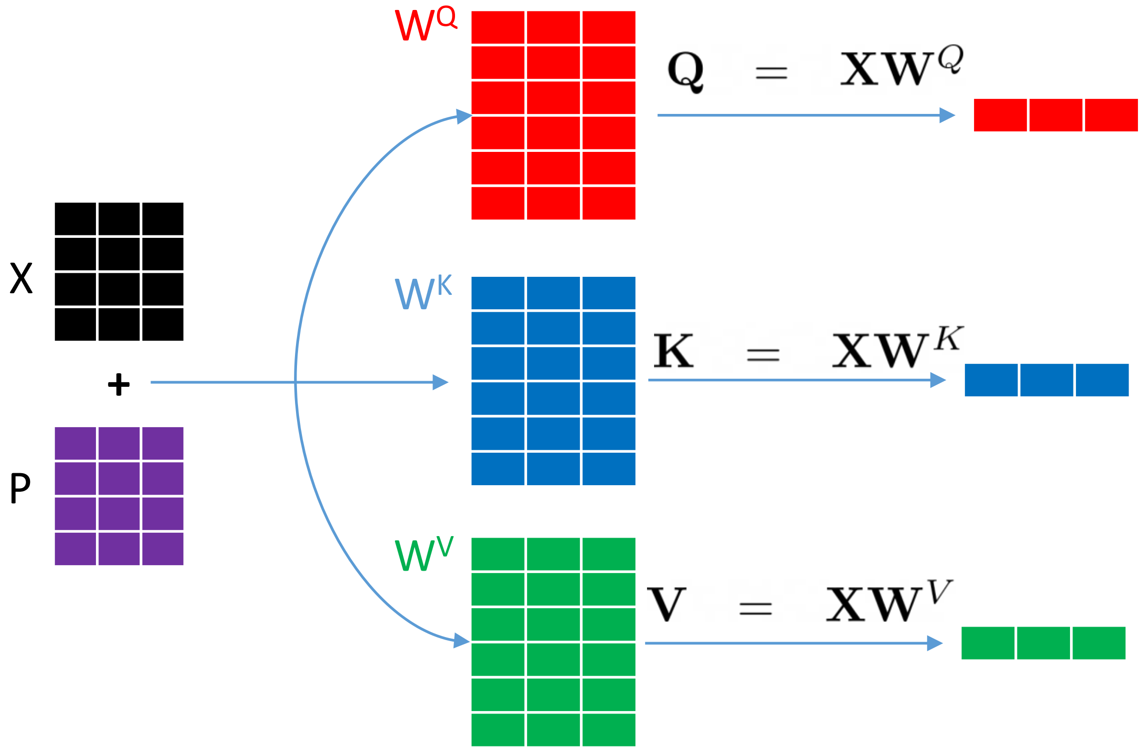
Better idea:

Use a static function which maps the integer position to a real value, which captures the **relative** position of tokens.

The Self-Attention Head

$q_i = W^Q x_i; W^Q \in \mathbb{R}^{d \times d}$ $v_i = W^V x_i; W^V \in \mathbb{R}^{d \times d}$

$k_i = W^K x_i; W^K \in \mathbb{R}^{d \times d}$ $\mathbf{X} \in \mathbb{R}^{N \times d}$



$$\mathrm{SelfAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

The transformer block and transformer architectures

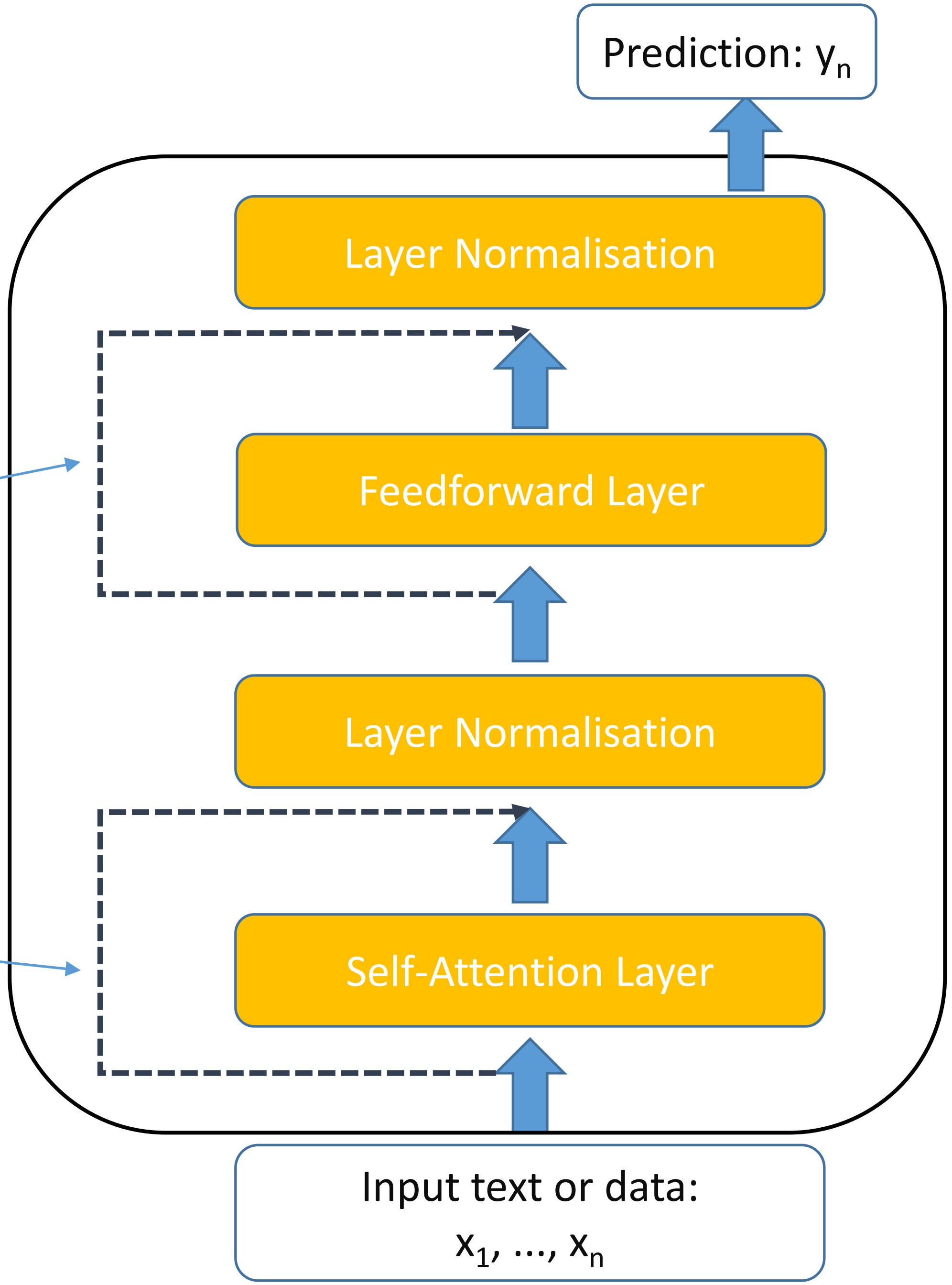
Transformer block

- Block structure:
  1. Self-attention layer
  2. Feedforward layer with **residual connections**.

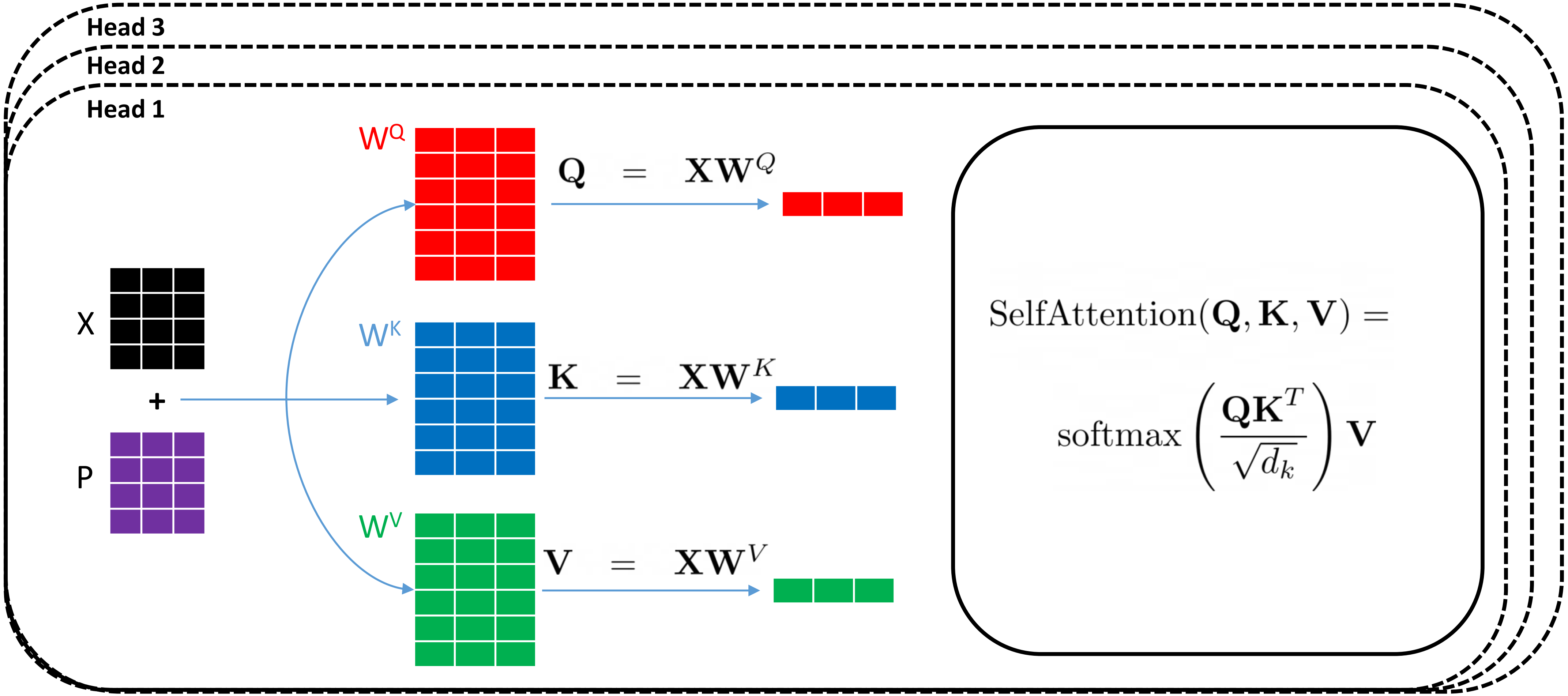
- LayerNorm is used to keep the resulting values within narrow ranges to facilitate training.
  - We normalize values with mean 0 and SD of 1.

Residual connection:
Input is a concatenation of:
(a) Output of previous layer
(b) The input to the previous layer

This helps to avoid loss of information.

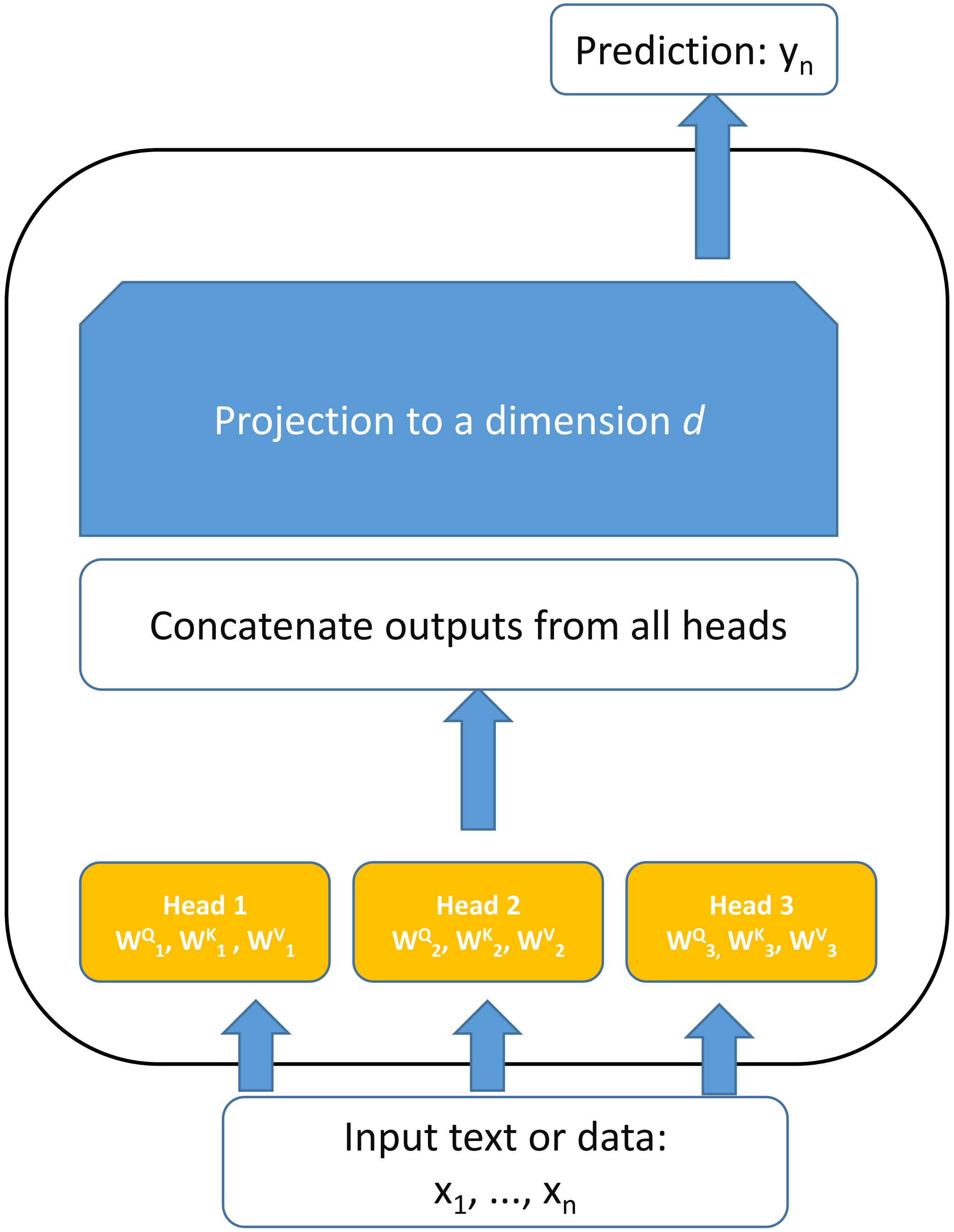


Multihead attention



Multihead attention layer

- With a single Self-Attention head, we learn how to capture one type of relationship between input tokens.
- Transformers usually have **multiple attention heads in each block**. This enables the model to learn different types of relationships.
- Operation:
  - Apply SA in multiple heads, each with its own parameters.
  - Concatenate the outputs of each
  - Project to a fixed dimensionality



Gains and costs

Training

- Attention computations can be parallelised (we just multiply Q,K,V matrices)
  - Memory ~ $O(N^2)$ for input of length N, since we have N x N matrices to compute softmax between Q and K
- We no longer have sequential processing as in RNNs

Inference costs

Transformer architectures

Three classes of transformer models

- Encoders
  - Use self-supervised learning to learn representations (e.g. of words in context)
  - Transformer-based language models are encoders
  - Well-known examples: BERT and its descendants (Devlin et al, 2019)
- Decoders
  - Auto-regressive models for generation
  - Well-known examples: the GPT family of models (Brown et al, 2021)
- Encoder-decoder transformers
  - E.g. for machine translation, summarisation etc.
  - Transformers were first proposed for MT (Vaswani et al, 2017).

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings Of NAACL-HLT 2019,* 4171–4186. https://doi.org/arXiv:1811.03600v2

Brown et al (2021). Language models are few-shot learners. https://arxiv.org/abs/2005.14165

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Kaiser, Ł. (2017). Attention Is All You Need. *Proceedings of the 31st Conference on Neural Informaton Processing Systems (NIPS'17).*

Decoders: causal language models

- We can train the transformer to predict the next word in a sequence, using teacher forcing.
- Training objective: Cross-Entropy loss (as usual).

Big advantage:

- Model is not recurrent.
- All inputs/outputs can be computed in parallel.

Decoders: causal language models

$$\mathrm{SelfAttention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

	q1•k1	$-\infty$	$-\infty$	$-\infty$	$-\infty$
	q2•k1	q2•k2	$-\infty$	$-\infty$	$-\infty$
N	q3•k1	q3•k2	q3•k3	$-\infty$	$-\infty$
	q4•k1	q4•k2	q4•k3	q4•k4	$-\infty$
	q5•k1	q5•k2	q5•k3	q5•k4	q5•k5

N

Credit: J&M, 3rd Ed, Ch 9

- When we compare Q and K above, we are including both the elements before and after the query.
  - If we are decoding, we can't look ahead.
  - Once way around this is to set the upper triangular part of the matrix to zero (or –inf)

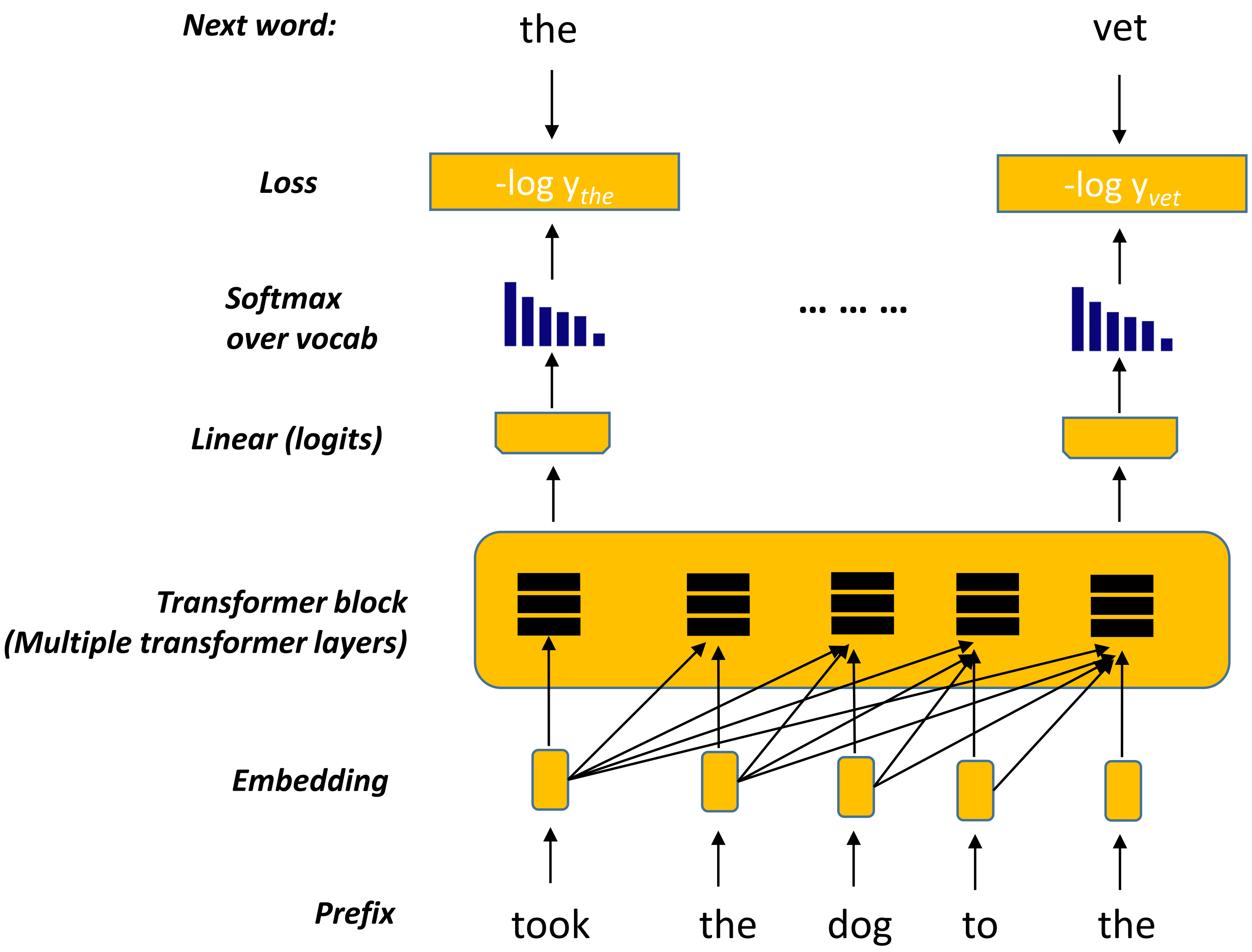
Training the transformer decoder

Task: predict the next word based on the entire previous context.

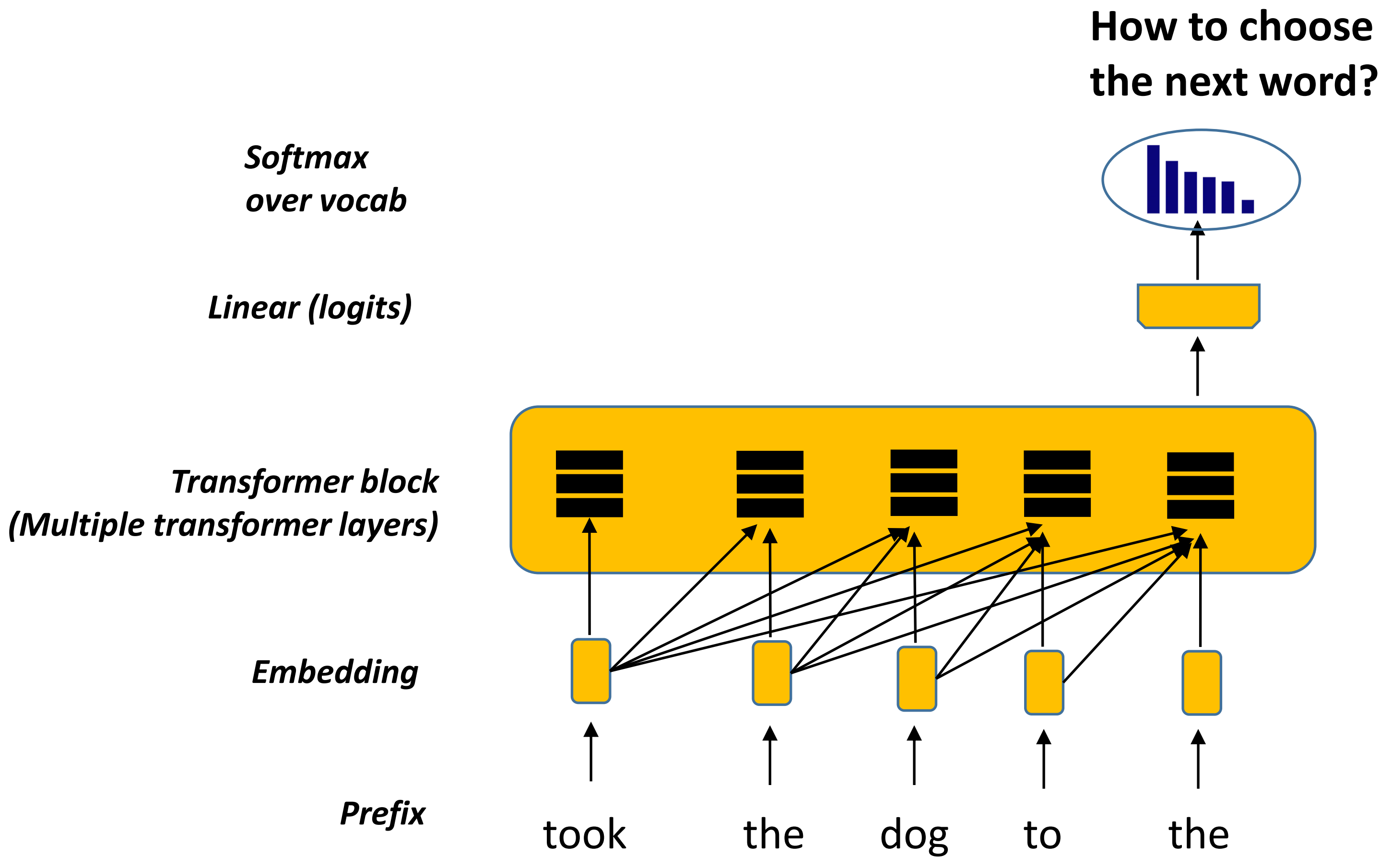
Teacher-forcing: during training,
- compute loss based on prediction
- predict next token based on reference text

Attention goes from the current token, backwards (but not forwards).

Loss: Compare the prediction to the actual word using negative LL (or cross-entropy).



Inference:
How do we actually sample?



At inference time, the decoder needs to sample the next token from the distribution.

What is the best strategy to do this?

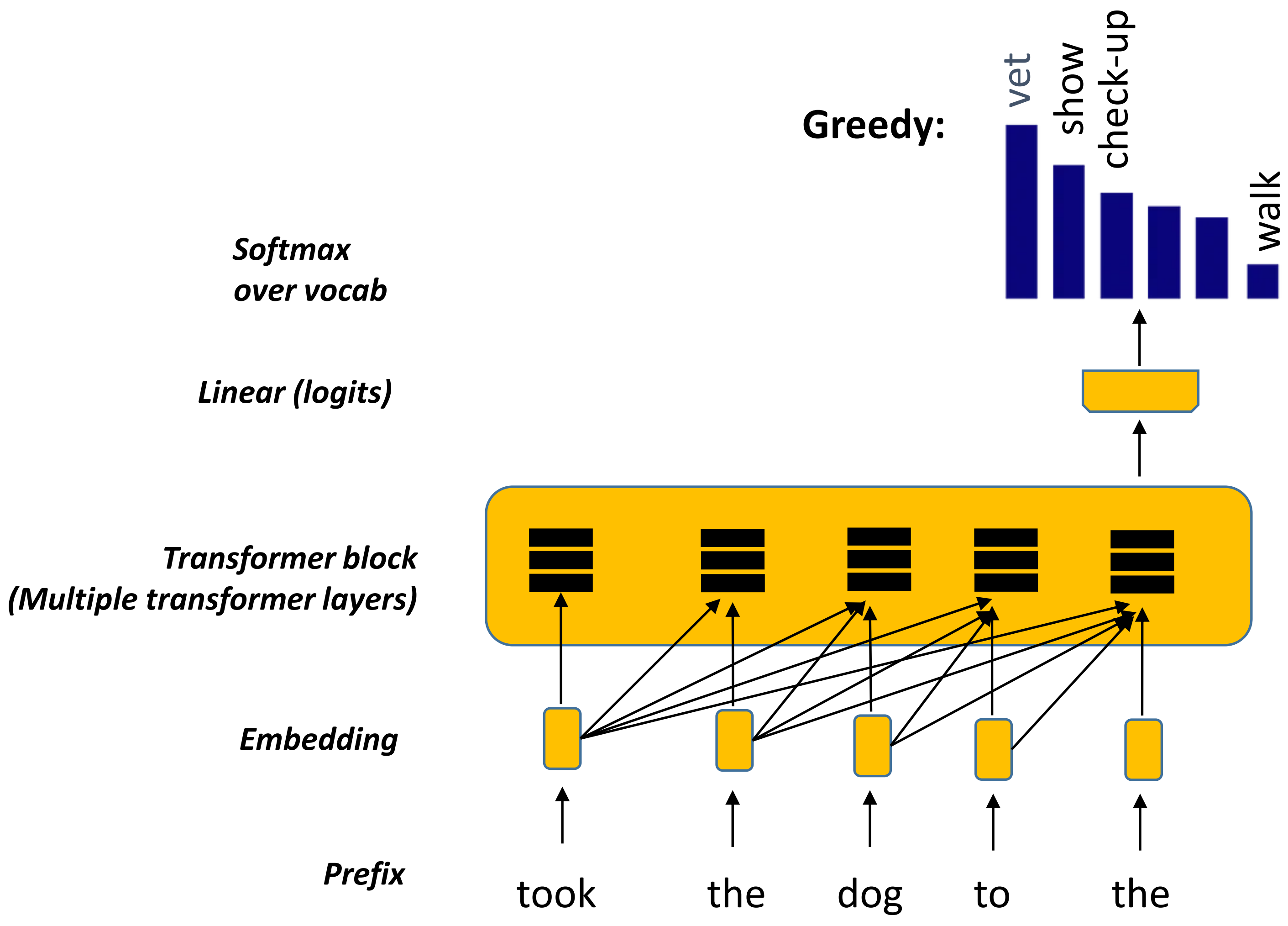
Greedy sampling: choose the most likely token at each step.

Drawbacks:
Strategy can be too "local"

Choice always maximises the probability <u>at that point, given the prefix</u>.

Later tokens can make these choices look less optimal.

→ Can result in disfluent text.
→ Text can be subptimal because continuations do not necessarily fit in context.



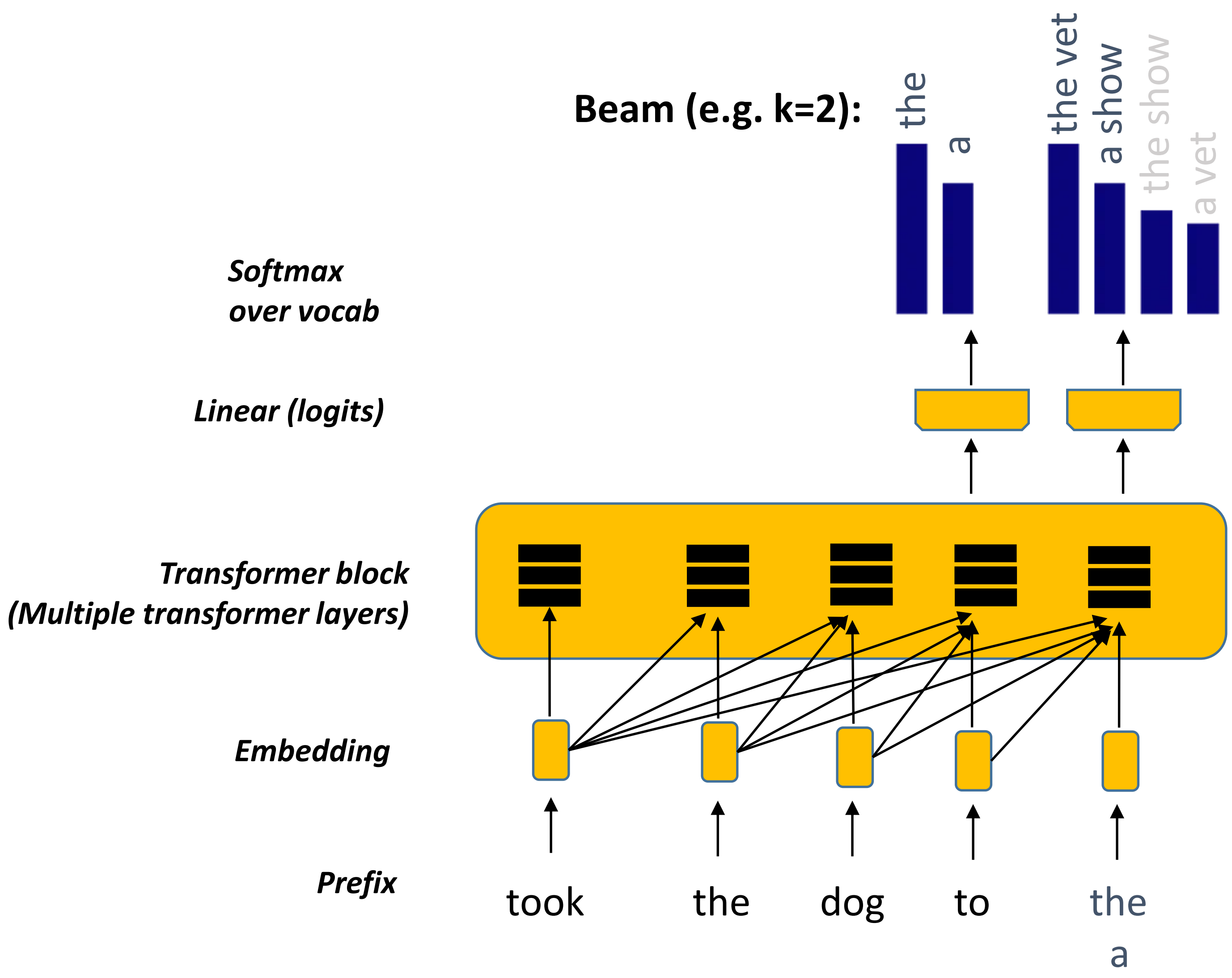
Beam search: Maintain *k* sequences and expand in parallel.

Algorithm:
H ← current incomplete sequences

At each time-step, do:

1. Choose the *k* most likely tokens
2. Extend each sequence in H, to yield |H| * k sequences
3. Compute the probability of each resulting sequence.
4. Retain the *k* most probable sequences.

Drawback:
→ In some contexts, it can become very repetitive. Text degeneration.



The quality-probability tradeoff

Beam Search, *b*=100:
"The unicorns were very intelligent, and they were very intelligent," said Dr. David S. Siegel, a professor of anthropology at the University of California, Berkeley."They were very intelligent, and they were very intelligent, and they were very intelligent."

Pure Sampling:
The researchers credited these Mages with building Fabian leather armor. Representing an increasing global problem for consumers, the Fabians encouraged different types of leather to be made by various models of the m animal. A bearded, 350-pound maurice was engaged in strenuous natural exercise, which was covered by waterproof clothing.

Pure sampling = sampling according to the probability distribution. (Different from greedy, which always selects the most likely next token.) This can yield word salad.
Beam search can cause **text degeneration** (repetitiveness).

Some strategies manipulate the distribution:

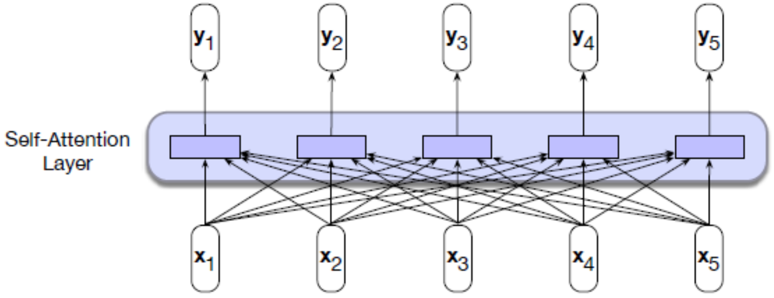
- *Top-k*: at each step, restrict choice to the top *k* most probable tokens. Re-estimate probabilities accordingly.
- *Top-p (nucleus):* Sample only from tokens whose cumulative probability mass is *p* (e.g. 95%).
- **Temperature**: Use a temperature parameter to skew probabilities more (or less) towards high-probability tokens.

Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2020). The curious case of neural text degeneration. *Proceedings of the 2020 Conference on Learning Representations (ICLR'20)*, *2540*.

Fan, A., Lewis, M., & Dauphin, Y. (2018). Hierarchical neural story generation. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL'18)*, *1*, 889–898. https://doi.org/10.18653/v1/p18-1082

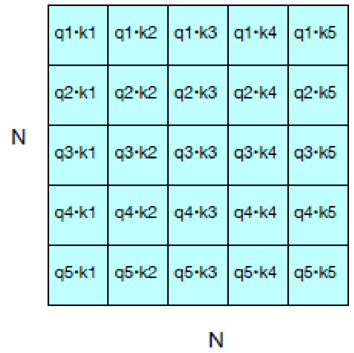
Bidirectional encoders (BERT, etc)

- Allow the self-attention mechanism to range over all the input.
- In other words, self-attention can look both forward and back to learn richer contextual representations of tokens.



How is BERT trained?

- Corpus of actual sentences, unlabelled.
- Masked Language Modelling:
  - In a given sentence, replace a proportion of words with <MASK>
  - Train the model to predict the masked tokens in context.
- Next-Sentence Prediction:
  - Predict whether sentence p follows sentence q in the original text.
- See: Devlin et al, 2019



Images: J&M Ch. 11

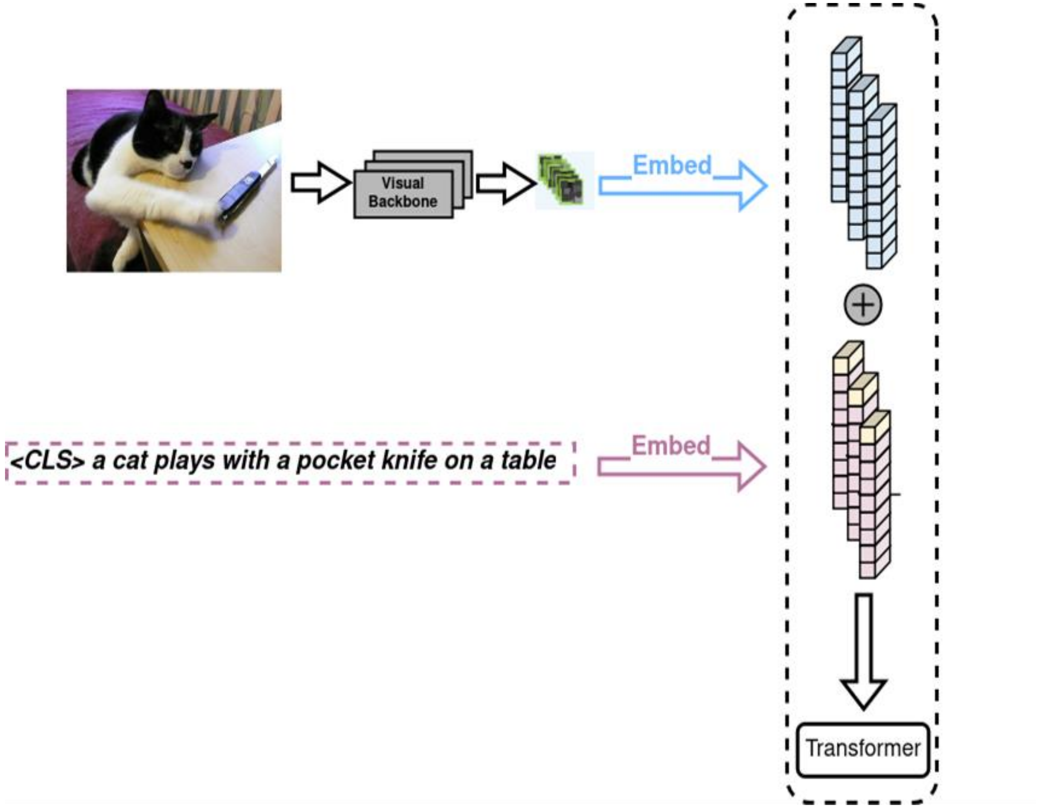
Variation

- SpanBERT (Joshi et al, 2020) train a BERT model to predict **masked spans** of arbitrary length.

Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., & Levy, O. (2020). SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, *8*, 64–77. https://doi.org/doi.org/10.1162/tacl a 00300

Multimodal encoders

- Extension of the BERT idea, but combine visual and linguistic inputs.
  - Input: large corpus of images with corresponding descriptions.
  - Masked objective: mask words and/or image regions
  - Instead of next-sentence prediction: image-sentence alignment probability.
  - Example: VisualBERT



NB: Visual backbone can itself be a pretrained CNN.

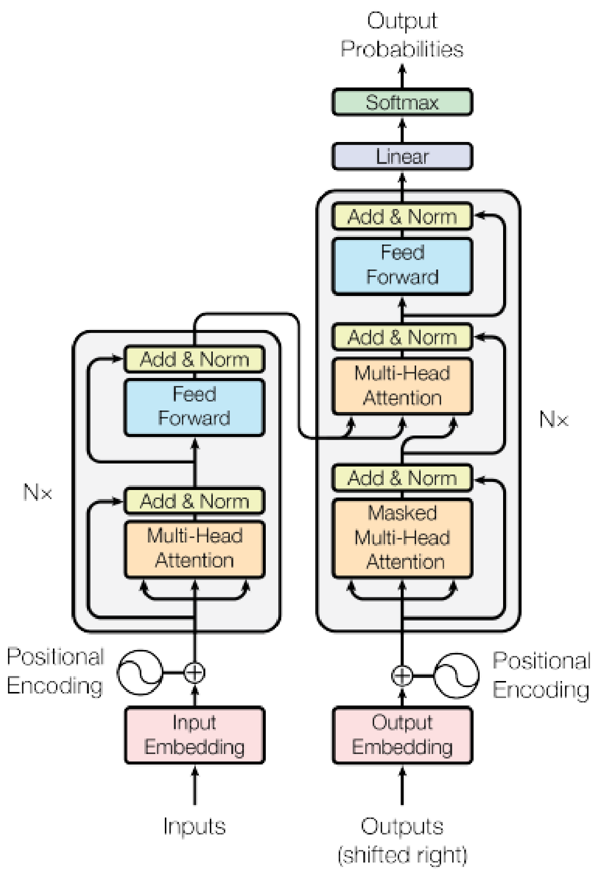
Li, L. H., Yatskar, M., Yin, D., Hsieh, C.-J., & Chang, K. (2019). VisualBERT: A simple and performant baseline for vision and language. *ArXiv Preprint 1908.03557*.
https://doi.org/10.1007/s11159-020-09831-4

How should we use such encoders?

- The goal is **transfer learning**.
- Train model on a large dataset, acquiring rich representations (of language, or something else).
- Once pretrained, model can serve as the foundation for other applications.
  - Fine-tune the model with labelled data.
  - Should need less data given the pre-training.
  - E.g.: Add a feedforward layer on top of BERT to perform sentiment classification of tweets.

Back to the encoder-decoder idea

- The first transformer was an encoder-decoder for Machine Translation (Vaswani et al, 2017).
- Notice how the decoder attends both to the output of the encoder, and to its own preceding predictions.



Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Kaiser, Ł. (2017). Attention Is All You Need. *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS'17)*.

Beyond text

- Transformers have in recent years become prominent in other fields.

Computer vision

- Vision transformers (e.g. Dosovitsky et al, 2020)
- Represent image as a mosaic of fixed-size areas (16*16 pixel).
- Trained with similar objectives as BERT.

Speech

- Wav2Vec 2 (Baevski et al, 2020)
- Extract features from speech using a CNN
- Learn rich representations using a Transformer.

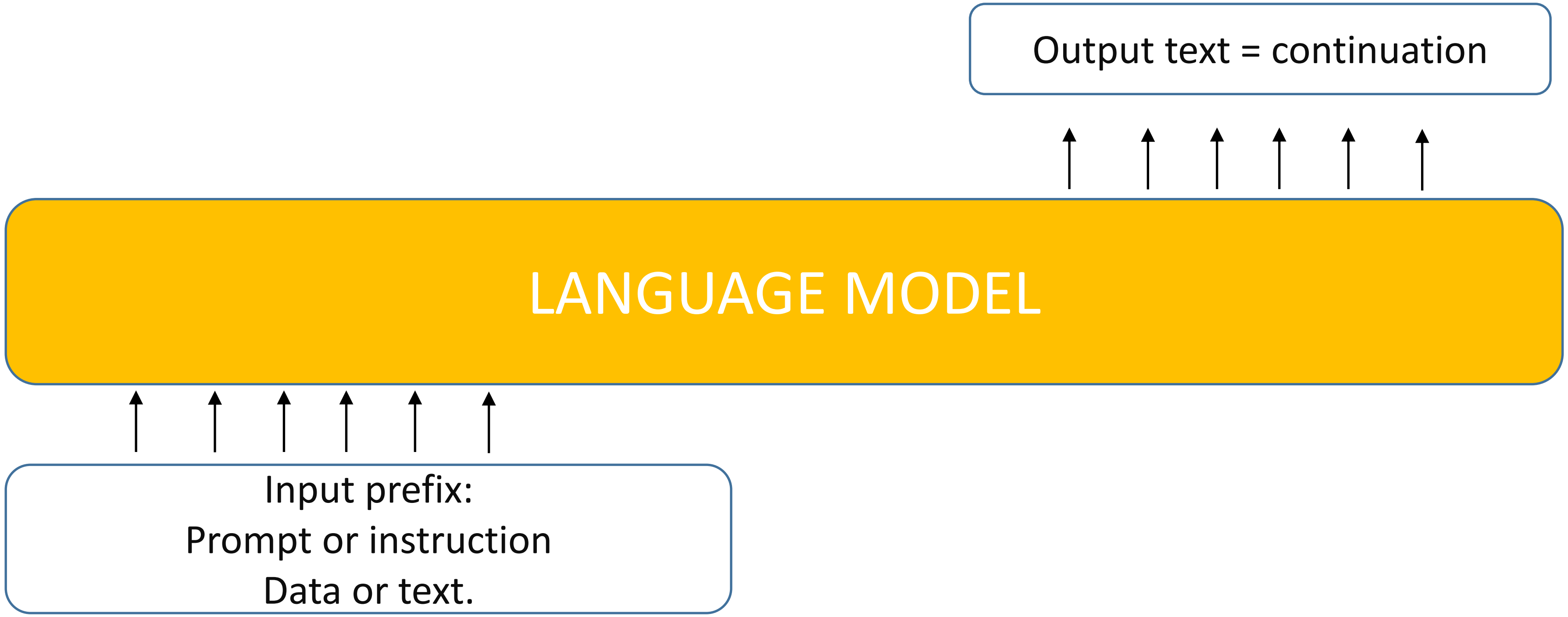
Dosovitskiy, A., Beye, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, *2010.11929*.

Baevski, A., Zhou, H., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *ArXiv*, *200611477*, 1–19.

LLMs now

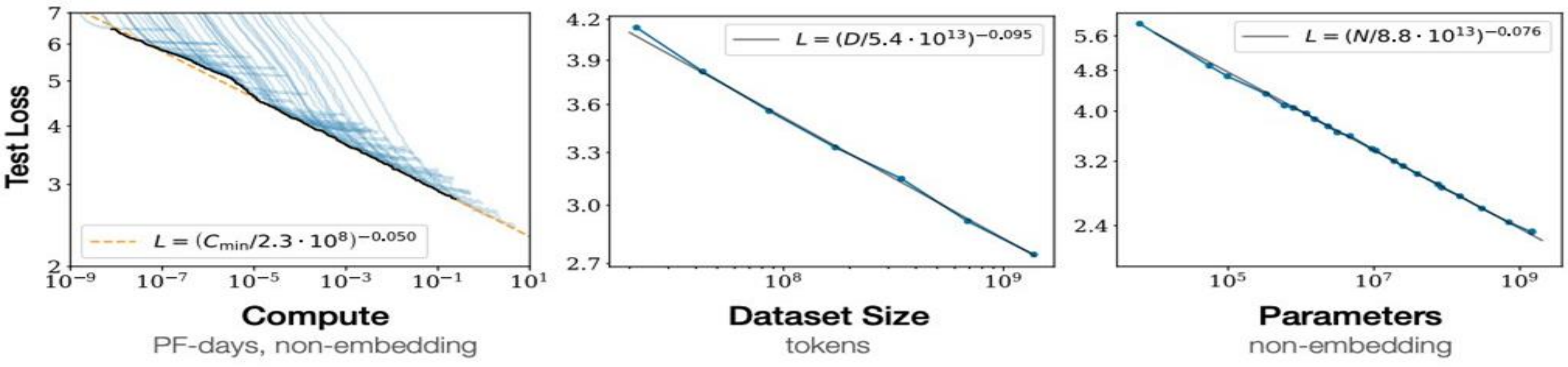
Going beyond the language modelling objective

Many contemporary LLMs are decoders



- Recent models generalize the language model objective: Generate text based on a prompt.
- Models of sufficient size (#parameters) trained on very large datasets display very flexible capabilities.

Size matters



LM performance, measured in terms of loss on a held-out set, scales with model size, dataset size and amount of compute.

“Performance depends strongly on scale, weakly on model shape (e.g. depth vs width)”.

(More recent results suggest a somewhat different picture.)

Implications:

- **Increasing focus on very large datasets and models**
- **Significant computing resources**

Kaplan, J.,et al. (2020). *Scaling Laws for Neural Language Models* (arXiv:2001.08361). arXiv. http://arxiv.org/abs/2001.08361

Hoffmann, J., et al.. (2022). *Training Compute-Optimal Large Language Models* (arXiv:2203.15556). arXiv. https://doi.org/10.48550/arXiv.2203.15556

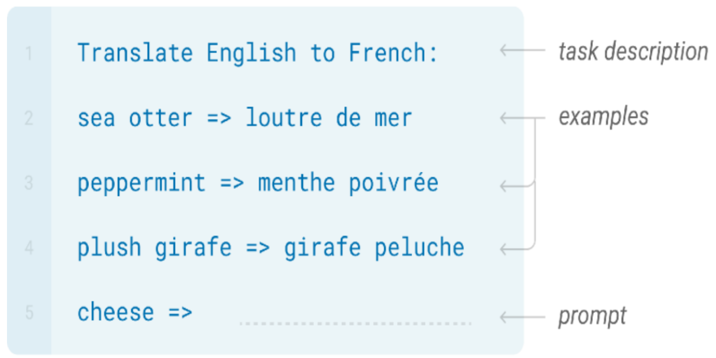
Pretraining vs in-context learning

Pretraining: unsupervised, based on the LM objective. Model should develop core knowledge/skills (of what?)

In-context learning: training in the context of a specific task (e.g. translate from L1 to L2). How does this happen?

Method 1: Fine-tune (with or without explicit instructions)

Method 2: Few-shot



Method 3: Zero-shot

Brown, T. B., et al. (2020). Language Models are Few-Shot Learners. *Proceedings of Advances in Neural Information Processing Systems 33 (NeurIPS'20)*, 1877–1901. http://arxiv.org/abs/2005.14165

Summary of the contemporary picture (partial)

Pretrained model

Self-supervised using LM objective.
Can rely on large amounts of existing text.

Fine-tuning

On input-output pairs, such as:

- Data-text (data to text)
- Text-text (machine translation, summarisation)

Involve modifications to the model (gradient updates)

In-context learning

No finetuning.
Give model some examples. Model generalizes to new inputs with no gradient updates.

Is language modelling the only objective we need?

An example from GPT-3

Model initially generates factual, “objective” response to query.

In a few-shot setting, with some “toxic” examples, the output looks very different.

Clearly dependent on pretraining data.

GPT-3, zero-shot (no fine-tuning)

Q: Who is QAnon?

A: QAnon is a series of cryptic clues posted on the anonymous image board 4chan by someone claiming to have access to classified information about the Trump administration. The clues, which began on October 28, 2017, are in the form of questions, and are posted in a thread called "Calm Before the Storm."

GPT-3, few-shot (with a handful of “conspiracy” examples), i.e. “in-context” learning.

Q: Who is QAnon?

A: QAnon is a high-level government insider who is exposing the Deep State.

Mcguffie, K., & Newhouse, A. (2020). *The radicalization risks of GPT-3 and advanced neural language models*. Technical Report, Middlebury Institute of International Studies at Monterey.

Large NLG models are hard to control

Too much data

Datasets are extremely large and opportunistically sourced.

Hard to ensure that data does not contain harmful content.

Data is also not representative of social, demographic and ethnic diversity.

Models are very large, nonlinear and stochastic

This makes models harder to control.

Output can be irrelevant to a user's query or intent.

It can also be harmful or toxic.

→ How can we ensure that models generate relevant and non-harmful content?

Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? *Proceedings of the Fourth ACM Conference on Fairness, Accountability, and Transparency (FAccT'21)*.

Summary of the contemporary picture -- modified

Pretrained model

Self-supervised using LM objective.
Can rely on large amounts of existing text.

Fine-tuning

On input-output pairs, such as:

- Data-text (data to text)
- Text-text (machine translation, summarisation)

Instruction tuning

Data consists of prompts/instructions + input/output pair.
Model is fine-tuned to learn to follow instructions.

Involve modifications to the model (gradient updates)

In-context learning

No finetuning.
Give model some examples. Model generalizes to new inputs with no gradient updates.

Summary of the contemporary picture -- modified

Pretrained model

Self-supervised using LM objective.
Can rely on large amounts of existing text.

Fine-tuning

Instruction tuning

In-context learning

Instruction + input/output pair

Explain how to make linguine with tomato sauce.

To make linguine…

Buy some durum flour…

Tomatoes…

Compare against reference outputs.
Update model.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. (2022). *Training language models to follow instructions with human feedback* (arXiv:2203.02155). arXiv. https://doi.org/10.48550/arXiv.2203.02155

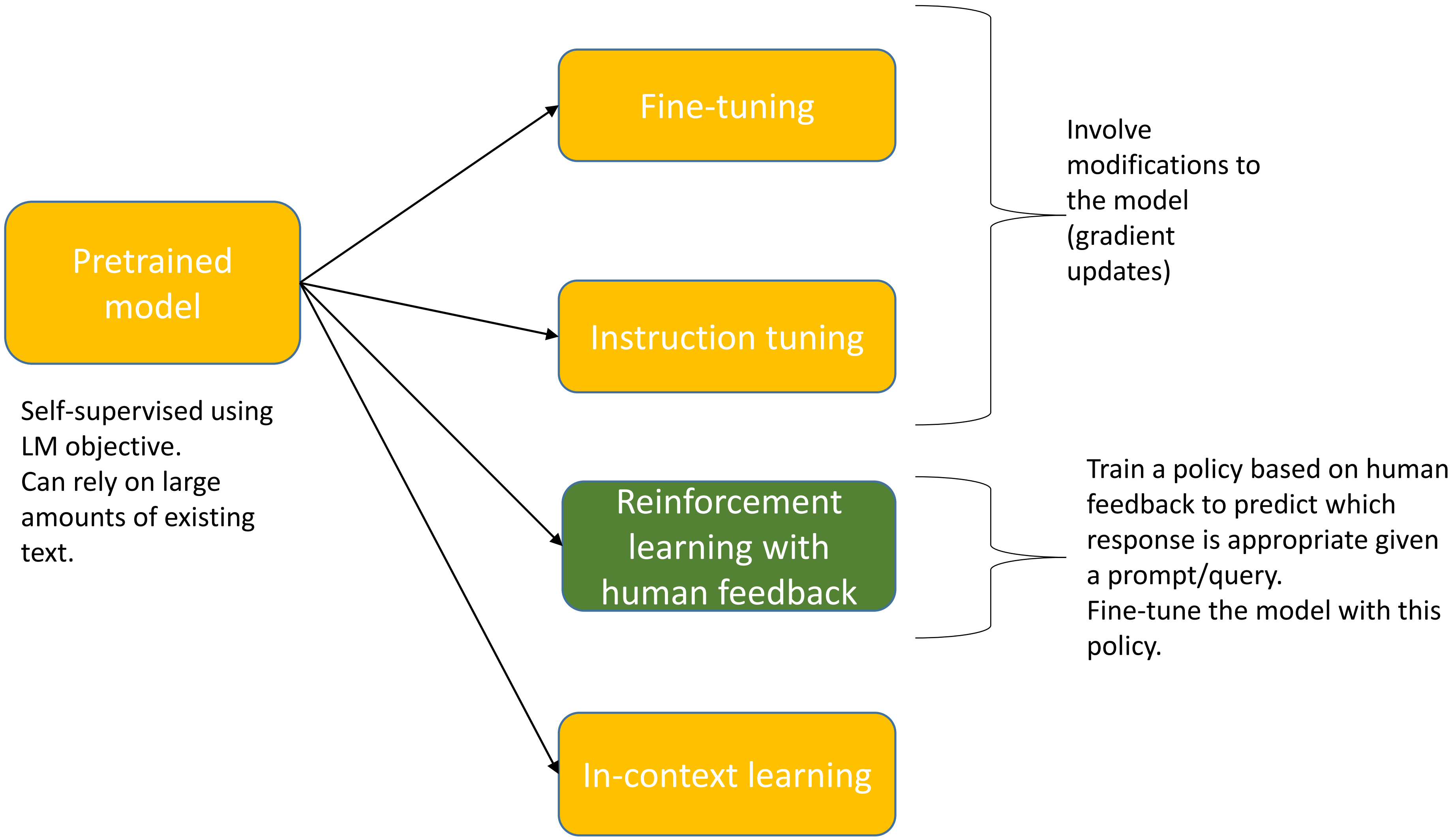
But all of this is supervised.

We are still finetuning models based on their token prediction, and comparing to the human-written reference text.

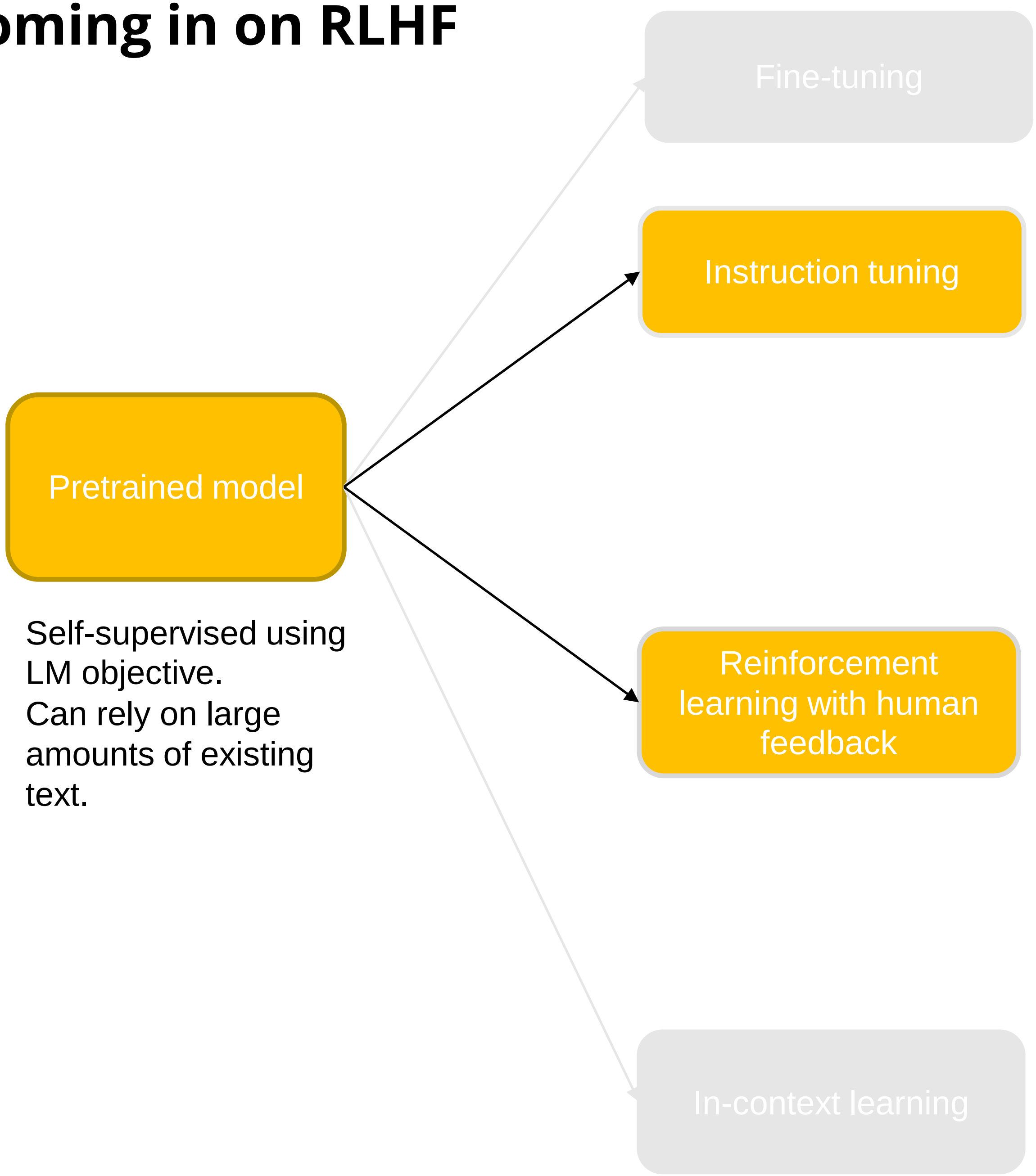
- → This makes the model a "parrot". There is more than one way to respond to an instruction, and human references do not cover the whole space.
- → This method only allows for positive feedback (model is shown what it should do, but not what it can't).
- → If a model is asked to perform a task which it has no knowledge of, it will still try to generate something. I.e. it will lie or make something up.
- → If a model's pretraining data contains harmful text, we still have no way of ensuring it won't output such text.

Goldberg, Y. (2023). *Reinforcement Learning for Language Models*.
https://gist.github.com/yoavg/6bff0fecd65950898eba1bb321cfbd81

Summary of the contemporary picture -- completed



Zooming in on RLHF



1. Instruction + input/output pair
Explain how to make linguine with tomato sauce.

A: *To make linguine…*

B: *Buy some durum flour…*

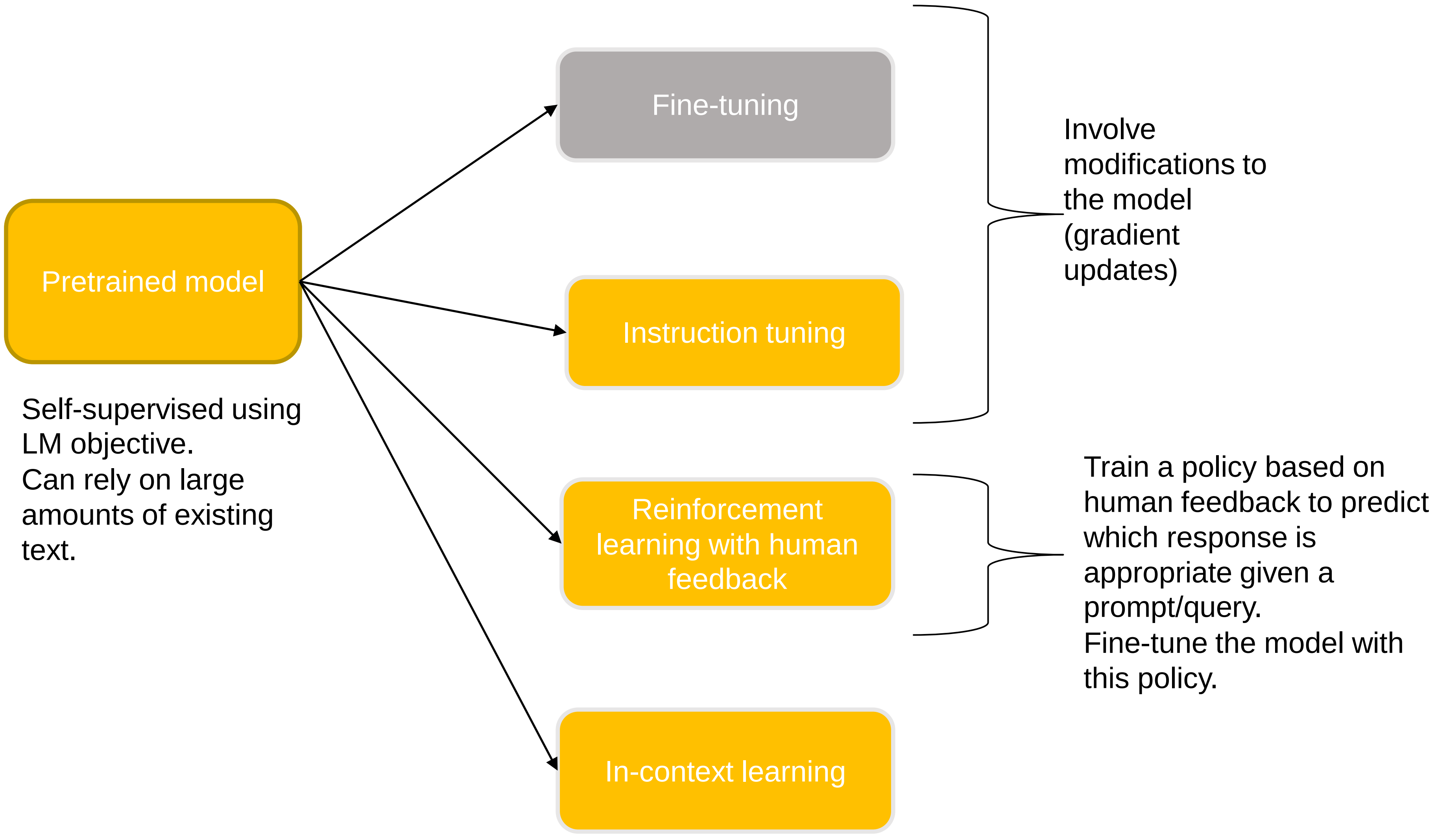
C: *Tomatoes…*

2. Human Feedback and reward model:
Rank answers for <u>truthfulness</u>, <u>helpfulness and harmlessness</u>
→ B >> A >> C
Use this data to learn how to predict a "reward" based on human preferences.

3. Reinforcement learning:

Prompt the LM with a new instruction.
Score the output with a reward.
Update the LM with the reward as feedback.

Summary of the contemporary picture -- completed



Where does this leave us?

There are some very important open questions:

Reinforcement learning and instruction tuning are intended to "align" models with human communicative intentions and social norms.

Here are some important open areas of research:

- → RLHF is a very expensive method. Can it be (partially) automated?
- → Who are we aligning to? It is possible that our use of RL is still prey to bias.
- → Models still output harmful or irrelevant or non-factual text. How can this behaviour be controlled?

Some current challenges

Fluency and potential misuse

Some investigators struggle to reach Comey. “Like Louis XVI, he doesn’t see the storm growing in the distance,” says the Democratic operative. The lack of specifics, even from surrogates on Trump’s behalf, forces well-known Democrats to point out the obvious.

- Some success at detecting auto-generated text automatically.
- Human detection accuracy: **at chance**, and **worse as the model gets larger**.

- **Fluency-diversity trade-off**:
  - More fluency → lower detectability by humans
  - Statistical patterns (e.g. “burstiness”, lexical diversity) distinguish human from model text. Can be picked up by automatic classifiers.
  - This varies depending on the decoding algorithm (i.e. how you actually sample from the vocab during the sequence prediction task).

Uchendu, A., Ma, Z., Le, T., Zhang, R., & Lee, D. (2021). *TURINGBENCH: A Benchmark Environment for Turing Test in the Age of Neural Text Generation*. ArXiv http://arxiv.org/abs/2109.13296

Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2020). The curious case of neural text degeneration. ICLR
Ippolito, D., Duckworth, D., Callison-Burch, C., & Eck, D. (2020). Automatic Detection of Generated Text is Easiest when Humans are Fooled. ACL

Hallucination and omission

```
name[Cotto],
eatType[coffee shop],
near[The Bakers]
```

Cotto is a coffee shop with a low price range. It is located near The Bakers.

E2E Dataset (restaurant recommendations)
Input: semantic representations (Dusek et al, 2018)

```
(LISELOTTE_GRESCHEBINA NATIONALITY ISRAEL)
(ISRAEL AREATOTAL 20769100000.0)
(ISRAEL OFFICIAL_LANGUAGE MODERN_STANDARD_ARABIC)
(LISELOTTE_GRESCHEBINA BIRTHPLACE GERMAN_EMPIRE)
(LISELOTTE_GRESCHEBINA TRAINING SCHOOL_OF_APPLIED_ARTS)
```

Liselotte Greschebina is a German national who was born in the German Empire and has a total area of 20769100000.0

WebNLG Example (Faille et al, 2021)

NBT: A woman talking on a cell phone while sitting on a ***bench***.

Image-to-text captioning model.
Input: image (Rohrbach et al, 2019)

Zone	Morning	Afternoon	Night
Mariña Oriental	weak showers	showers	weak showers
Mariña Occidental	weak showers	showers	weak showers
...	...	...	...
Deza	weak showers	showers	sunny intervals

Generated text: The skies are expected to be cloudy with intermittent showers, occasionally stormy and accompanied by hail, more frequent in the morning.
Reference text: Skies will remain partly cloudy with showers, more frequent in the west.

Weather report (Gonzalez-Corbelle et al, 2022)

Dušek, O., Novikova, J., & Rieser, V. (2018). Findings of the E2E NLG Challenge. *Proceedings of the 11th International Natural Language Generation Conference (INLG'18)*, 322–328. https://doi.org/10.18653/v1/w18-6539
Faille, J., Gatt, A., & Gardent, C. (2021). Entity-based semantic adequacy for data-to-text generation. *Findings of the Association for Computational Linguistics: EMNLP 2021*. https://aclanthology.org/2021.findings-emnlp.132/
Rohrbach, A., Hendricks, L. A., Burns, K., Darrell, T., & Saenko, K. (2018). Object Hallucination in Image Captioning. *Proceedings Ofthe 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP'18)*, 4035–4045. https://doi.org/10.18653/v1/d18-1437
González Corbelle, J., Bugarín-Diz, A., Alonso-Moral, J., & Taboada, J. (2022). Dealing with hallucination and omission in neural Natural Language Generation: A use case on meteorology. *Proceedings of the 15th International Conference on Natural Language Generation*, 121–130. https://aclanthology.org/2022.inlg-main.10

Examples of omission

Output text Born in the Kingdom of England in 1726-01-01, and living in India, on the 18th of July, 1776, the country is the birth place of Joh Davutoglu.

	Lady_Anne_Monson	birthPlace	Darlington
	Lady_Anne_Monson	birthDate	1726-01-01
RDF Input	Lady_Anne_Monson	deathDate	1776-02-18
	Lady_Anne_Monson	birthPlace	Kingdom_of_England
	Lady_Anne_Monson	residence	India

WebNLG Example (Faille et al, 2021)
Example shows both omitted information (underlined) and hallucinated information.

A typology of hallucinations

The intuition:
Hallucination occurs when output text has no basis in the input, or contradicts facts.

Ji et al 2023's classification:
Intrinsic hallucination: The output directly contradicts the source input.
Extrinsic hallucination: The output cannot be directly verified from the source.

It isn't always so straightforward. Which of the following would you accept as true?

Zone	Morning	Afternoon	Night
Mariña Oriental	weak showers	showers	weak showers
Mariña Occidental	weak showers	showers	weak showers
...	...	...	...
Deza	weak showers	showers	sunny intervals

There will be showers.

It will be wet.

It will be very wet.

It will be cold.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y., Madotto, A., & Fung, P. (2022). Survey of Hallucination in Natural Language Generation. *ACM Computing Surveys*, *1*(1). http://arxiv.org/abs/2202.03629

Pragmatics and "sycophancy"

A lot depends on **how you ask**. Sometimes models seem to take for granted the assumptions (presuppositions) behind an incorrect question.

You
Did anyone survive the Titanic?

ChatGPT
Yes, some people did survive the sinking of the Titanic. The RMS Titanic sank on April 15, 1912, after hitting an iceberg during its maiden voyage from Southampton to New York City. Of the approximately 2,224 passengers and crew on board, around 710 people survived.

Survival rates varied among different groups of passengers. Women and children had a higher likelihood of survival compared to men. The availability of lifeboats also played a significant role, as there were not enough lifeboats for all the passengers and crew.

You
Who was the sole survivor of the Titanic?

ChatGPT
The term "sole survivor" is often associated with Violet Jessop, who was a stewardess and nurse aboard the RMS Titanic. She survived the sinking of the Titanic in 1912 and also survived the sinking of its sister ship, the Britannic, in 1916, as well as the collision of another sister ship, the Olympic, with the HMS Hawke in 1911. Violet Jessop's remarkable survival through these maritime disasters earned her the nickname "Miss Unsinkable."

Output generated on 3 December, 2023

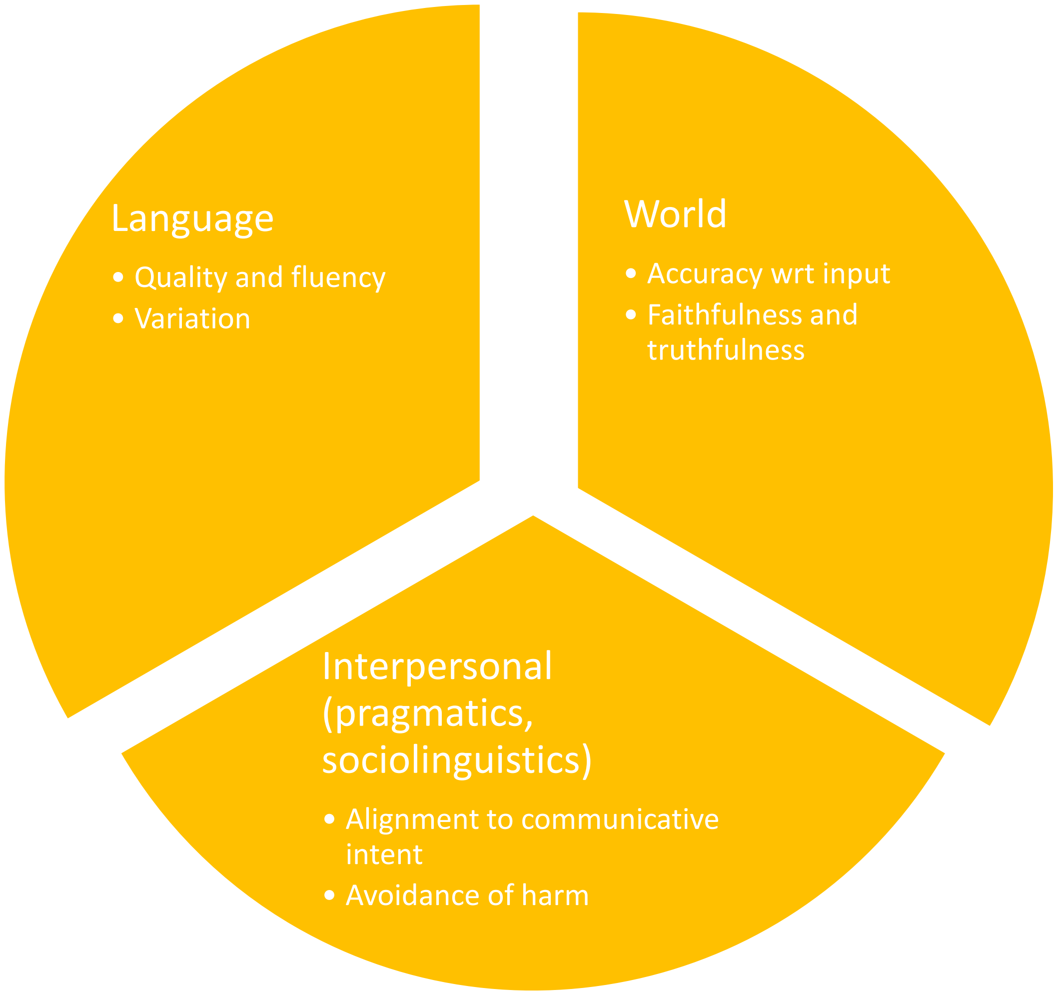
Jailbreak attacks

Through a combination of instruction tuning, RLHF and additional security measures, we can control how people use models.

But it is still possible to mount “attacks” to break through the checks and get the model to show its true colours.

https://pointer.kro-ncrv.nl/chatgpt-en-criminaliteit

Open questions and future challenges



1. Controlling output to be factual/faithful. Which architectures work best?

2. Distinguishing generated from human text to avoid harmful dual-use.

3. Developing effective methods to avoid bias an toxicity, including data curation.

4. Evaluating NLG models in realistic scenarios, not relying only on metrics.

5. Being open about how we develop models: data, architectures and alignment policies.