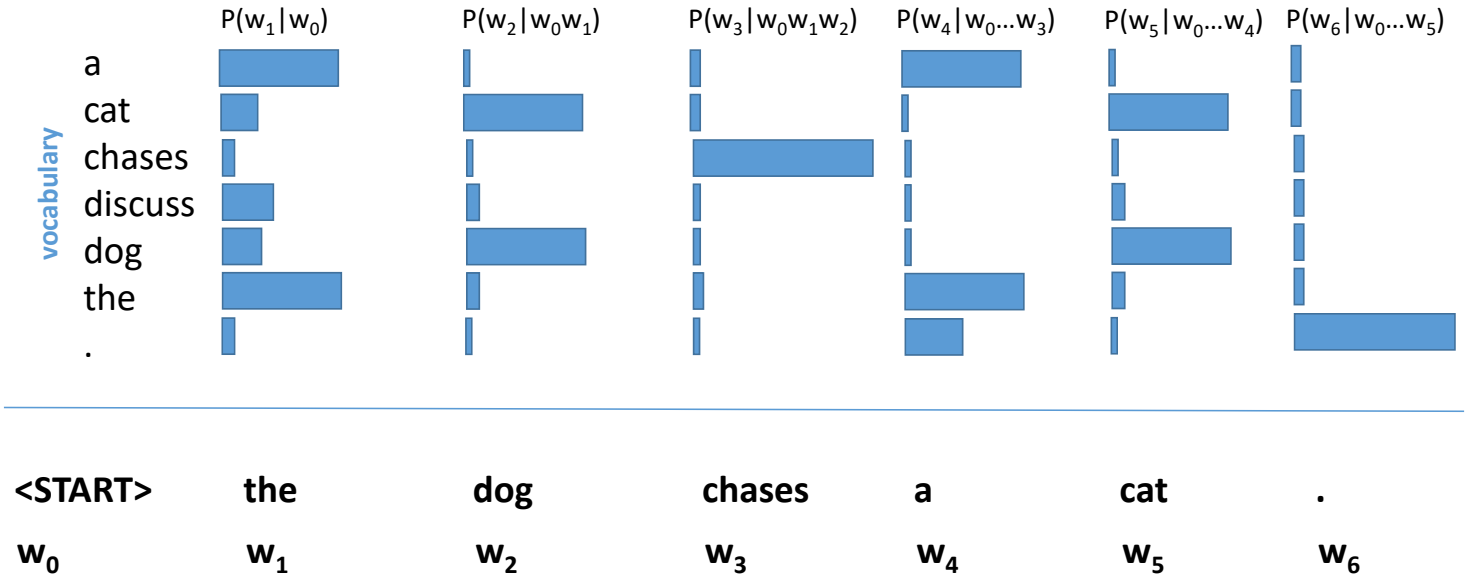
BERT

Measuring Performance

Learning goals

- Metrics for text output
- Task-specific metrics
- Task-agnostic model evaluation

ARLMS



<START>	the	dog	chases	a	cat	.
w_0	w_1	w_2	w_3	w_4	w_5	w_6

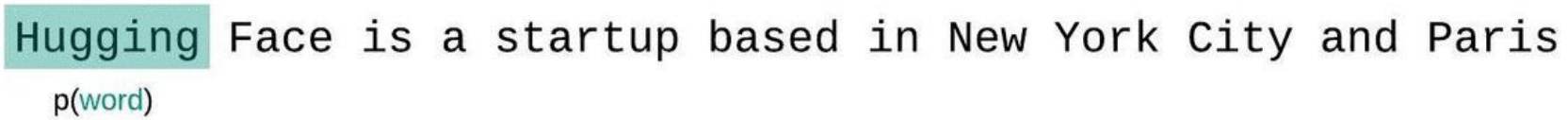
- **Question:** How can we measure “performance“ on this task?

ARLM PERPLEXITY (1)

- Well-defined for ARLMs; trickier for MLMs
- Intuitively: *Amount of the model's surprisal when confronted with a sequence* (higher value means higher "surprisal")
- More technically: Measure of uncertainty of a probabilistic model
- Example: Perplexity of a fair k-sided die (uniform distribution) is k

ARLM PERPLEXITY (2)

- Autoregressive factorization of the sequence:



▸ Source: huggingface

- Log-probability of i-th token given context: $\log(p_\theta(w_i|w_{<i}))$
  more "certain" model $\rightarrow$ higher log-probability
- Aggregate[1] over the whole sequence:

$$PPL = \exp\left(\frac{1}{t}\sum_{i=1}^{t}\log(p_\theta(w_i|w_{<i}))\right)$$

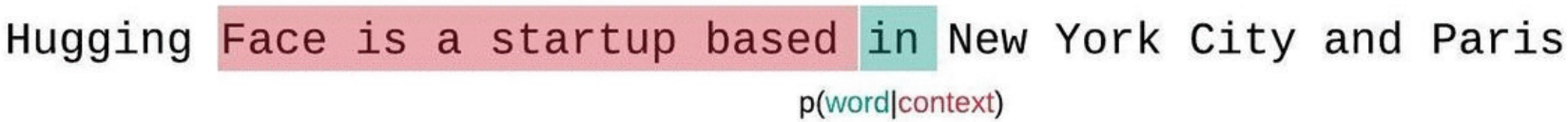
[1] The choice of the log's base is basically arbitrary.

ARLM PERPLEXITY (3)

- *Lower bound* is **1**, i.e. the model predicts *every* token correctly (with a certainty of 100%)
- **Question:** Is this really desirable?
- *Upper bound* is $|V|$, i.e. the model only provides a random guess for every token with a probability of $\frac{1}{|V|}$
- Selection of state-of-the-art perplexities:
  - **1B Word Benchmark** ▸ Chelba et al., 2013 ($|V| = 800k$)
    PPL = 21,8 ▸ Dai et al., 2019
  - **WikiText-103** ▸ Merity et al., 2016 ($|V| = 270$k)
    PPL = 10,8 ▸ Shoeybi et al., 2019
  - **Penn Treebank** ▸ Marcus et al., 1994 ($|V| = 10k$)
    PPL = 20,5 ▸ Brown et al., 2020

ARLM PERPLEXITY (4)

- *Problem:* Fixed context size of models (e.g. 1024 for GPT-2)



▸ Source: huggingface

- *Possible solution:* Sliding window strategy
- Close approximation to “true“ autoregressive decomposition
- *Drawback:* Computationally expensive (individual forward pass for each token)

EVALUATING GENERATED TEXT (1)

- **Question:** How can we evaluate the quality of generated text?
- *Use cases:*
  - Machine translation
  - Question answering (extractive or abstractive)
  - Dialogue generation
  - Text summarization
  - Image Captioning
  - Code generation

EVALUATING GENERATED TEXT (2)

Machine Translation

- Metrics based on N-gram-overlap
  - BLEU (cf. Chap. 3.1) ▸ Papineni et al., 2002
  - ROUGE ▸ Lin, 2004
  - METEOR ▸ Banerjee and Lavie, 2005
- Metrics based pre-trained (neural) models
  - BertScore ▸ Zhang et al., 2019
  - BLEURT ▸ Sellam et al., 2020
  - COMET ▸ Rei et al., 2020

EVALUATING GENERATED TEXT (3)

- BLEU, ROUGE, METEOR:
  - "Bilingual evaluation understudy", originally for machine translation
  - Measures overlap of words, and longer word sequences of different sizes
  - BLEU: unreliable per-example, but correlates with human judgments on an aggregate level ▸ Reiter, 2018
- BERTScore (based on pre-trained BERT model)
  - Also able to consider contextual similarities
  - Correlates better with human judgements
- BLEURT
  - Better per-example quality estimation by fine-tuning on examples with judgements
  - But: potential bias if tested systems are very different than training examples

EVALUATING GENERATED TEXT (4)

Question Answering / Summarization / Dialogues

- Aspects to consider
  - Factual correctness
  - Fluency
  - Stylistic aspects
  - Engagement
  - ...
- Human evaluation?!

TASK-SPECIFIC EVALUATION: CLASSIFICATION

(Binary) Document-level Classification

- Accuracy
- Recall / Precision
- F1-Score
- ROC-Curve / AUC
- ...

(Multi-Class) Document-level classification

- Micro- / Macro-averaged F1
- Class-specific accuracies
- ...

TASK-SPECIFIC EVALUATION: RANKING

Ranking:

- Use Cases:
  - Retrieval systems output a ranking of relevant documents
  - Search engines return top *k* results
  - ...
- Ranking evaluation metrics
  - Mean average precision (MAP)
  - Mean reciprocal rank (MRR)
  - Precision@k / Recall@k
  - ...

TASK-SPECIFIC EVALUATION: RANKING

MRR:

$$RR = \frac{1}{\text{rank of first relevant item}}$$

$$MRR = \frac{1}{Q} \sum_{q \in Q} RR_q, \ \textit{with } Q = \text{set of queries}$$

- Use Cases:
  - Retrieval systems output a ranking of relevant documents
  - Search engines return top *k* results
  - ...
- Ranking evaluation metrics
  - Mean average precision (MAP)
  - Mean reciprocal rank (MRR)
  - Precision@k / Recall@k
  - ...