Machine Translation

- Machine Translation (MT): The use of computers to translate from one language to another.
- Usage
  - Information access
  - Reduce/avoid digital divide
  - Computer Aided Translation (CAT)
  - Translating speech on-the-fly for communication

Challenges

- In machine translation, the words of the target language don’t necessarily agree with the words of the source language in number or order.
- English: He wrote a letter to a friend
- Japanese: tomodachi ni tegami-o kaita

  friend to letter wrote
- In English, the verb is in the middle of the sentence, while in Japanese, the verb kaita comes at the end. The Japanese sentence doesn’t require the pronoun he, while English does.

大会/General Assembly 在/on 1982年/1982 12月/December 10日/10 通过了/adopted 第37号/37th 决议/resolution， 核准了/approved 第二次/second 探索/exploration 及/and 和平peaceful 利用/using 外层空间/outer space 会议/conference 的/of 各项/various 建议/suggestions 。

On 10 December 1982 , the General Assembly adopted resolution 37 in which it endorsed the recommendations of the Second United Nations Conference on the Exploration and Peaceful Uses of Outer Space .

Note the many ways the English and Chinese differ. For example the ordering differs in major ways;
the Chinese order of the noun phrase is “peaceful using outer space conference of suggestions” while the English has “suggestions of the ...conference on peaceful use of outer space”.
And the order differs in minor ways (the date is ordered differently).
English requires *the* in many places that Chinese doesn’t,
and adds some details (like “in which” and “it”) that aren’t necessary in Chinese.
Chinese doesn’t grammatically mark plurality on nouns (unlike English, which has the “-s” in “recommendations”), and so the Chinese must use the modifier 各项/various to make it clear that there is not just one recommendation.

Language Divergences and Typology

- There are about 7,000 languages in the world.
- Some aspects of human language seem to be universal, holding true for every one of these languages
- Every language, for example, seems to have words for referring to people, for talking about eating and drinking, for being polite or not.
- There are also structural linguistic universals; for example, every language seems to have nouns and verbs, has ways to ask questions, or issue commands, has linguistic mechanisms for indicating agreement or disagreement.

- Languages also differ in many ways
- Understanding what causes such translation divergences can help us build better MT models.
- We often distinguish the idiosyncratic and lexical differences that must be dealt with one by one
- systematic differences that we can model in a general way (many languages put the verb before the grammatical object; others put the verb after the grammatical object)
- The study of these systematic cross-linguistic similarities and differences is called **linguistic typology**.

Word Order Typology

- German, French, English, and Mandarin, for example, are all SVO (Subject-Verb-Object) languages, meaning that the verb tends to come between the subject and object.
- Hindi and Japanese, by contrast, are SOV languages, meaning that the verb tends to come at the end of basic clauses.
- Irish and Arabic are VSO languages.
- Two languages that share their basic word order type often have other similarities. For example, VO languages generally have prepositions, whereas OV languages generally have postpositions.

- In this SVO English sentence, the verb wrote is followed by its object a letter and the prepositional phrase to a friend, in which the preposition to is followed by its argument a friend.
- Arabic, with a VSO order, also has the verb before the object and prepositions.
- By contrast, in the Japanese example that follows, each of these orderings is reversed; the verb is preceded by its arguments, and the postposition follows its argument.

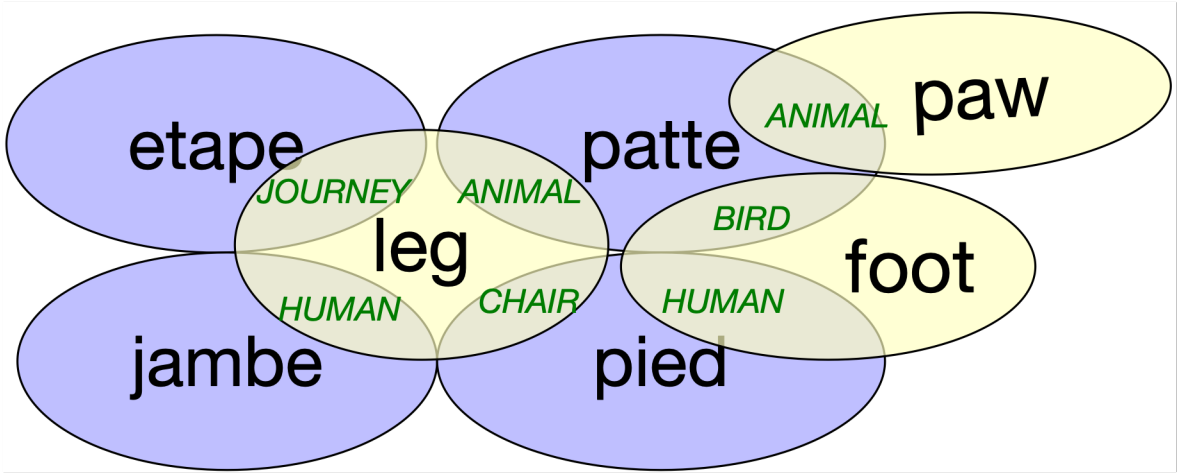
English: *He wrote a letter to a friend*

Japanese: *tomodachi ni tegami-o kaita*
friend to letter wrote

Arabic: *katabt risāla li ṡadq*
wrote letter to friend

Lexical Divergences

- We also need to translate the individual words from one language to another.
- For any translation, the appropriate word can vary depending on the context.



Lexical gap

- One language may have a lexical gap, where no word or phrase, short of an explanatory footnote, can express the exact meaning of a word in the other language.
- For example, English does not have a word that corresponds neatly to Mandarin xi$\grave{a}$o or Japanese oyak$\bar{o}$k$\bar{o}$

Morphological Typology and Referential density

- Morphologically, languages are often characterized along two dimensions of variation.
  - The first is the number of morphemes per word
  - The second dimension is the degree to which morphemes are segmentable
- Languages vary along a typological dimension related to the things they tend to omit
- Languages that can omit pronouns are called pro-drop languages.
- Even among the pro-drop languages, there are marked differences in frequencies of omission. Japanese and Chinese, for example, tend to omit far more than does Spanish.
- We say that languages that tend to use more pronouns are more referentially dense than those that use more zeros.

Machine Translation using Encoder-Decoder

- Given a sentence in a source language, the MT task is then to generate a corresponding sentence in a target language.
- For example, an MT system is given an English sentence like

  The green witch arrived

- Must translate it into the Spanish sentence:

  Lleg´ o la bruja verde

- MT uses supervised machine learning: at training time the system is given a large set of parallel sentences
- It learns to map source sentences into target sentences.
- In practice, rather than using words (as in the example above), we split the sentences into a sequence of subword tokens (tokens can be words, or subwords, or individual characters).
- The systems are then trained to maximize the probability of the sequence of tokens in the target language $y_1,...,y_m$ given the sequence of tokens in the source language $x_1,...,x_n$: $P(y_1,...,y_m|x_1,...,x_n)$

- Rather than use the input tokens directly, the encoder-decoder architecture consists of two components, an encoder and a decoder.
- The encoder takes the input words $x = [x_1,...,x_n]$ and produces an intermediate context h.
- At decoding time, the system takes h and, word by word, generates the output y:

  $h = \text{encoder}(x)$

  $y_{t+1} = \text{decoder}(h, y_1,...,y_t)) \; \forall t \in [1,...,m]$

Tokenization

- Machine translation systems use a vocabulary that is fixed in advance, and rather than using space-separated words, this vocabulary is generated with subword tokenization algorithms, like the BPE algorithm.
- Rather than the simple BPE algorith, modern systems often use more powerful tokenization algorithms. Some systems (like BERT) use a variant of BPE called the wordpiece algorithm, which instead of choosing the most frequent set of tokens to merge, chooses merges based on which one most increases the language model probability of the tokenization.

Wordpiece Algorithm

1. Initialize the wordpiece lexicon with characters (for example a subset of Uni-code characters, collapsing all the remaining characters to a special unknown character token).
2. Repeat until there are V wordpieces:

   (a) Train an n-gram language model on the training corpus, using the current set of wordpieces.

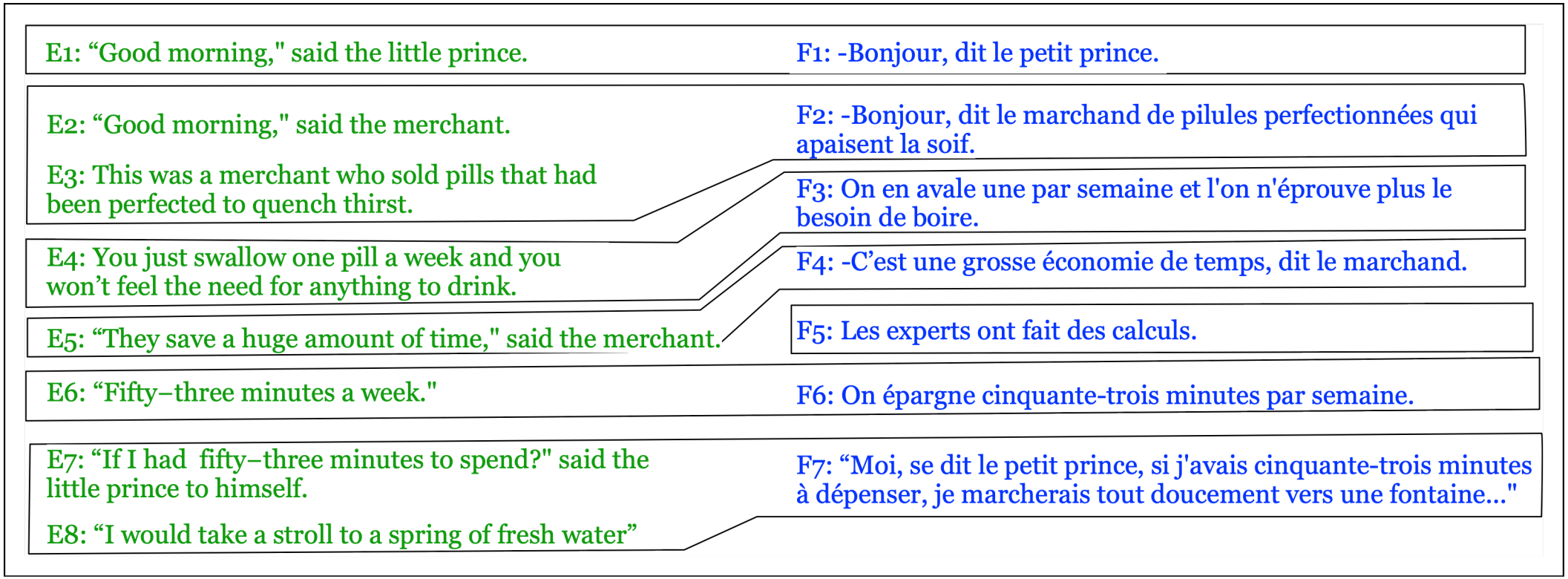
   (b) Consider the set of possible new wordpieces made by concatenating two wordpieces from the current lexicon. Choose the one new wordpiece that most increases the language model probability of the training corpus.

Creating the Training data

- Machine translation models are trained on a parallel corpus, sometimes called a bitext, a text that appears in two (or more) languages.
- Large numbers of parallel corpora are available.
  - Some are governmental; the Europarl corpus (Koehn, 2005), extracted from the proceedings of the European Parliament, contains between 400,000 and 2 million sentences each from 21 European languages.
  - The United Nations Parallel Corpus contains on the order of 10 million sentences in the six official languages of the United Nations (Arabic, Chinese, English, French, Russian, Spanish)

Sentence alignment

- Standard training corpora for MT come as aligned pairs of sentences.
- When creating new corpora, for example for under resourced languages or new domains, these sentence alignments must be created.

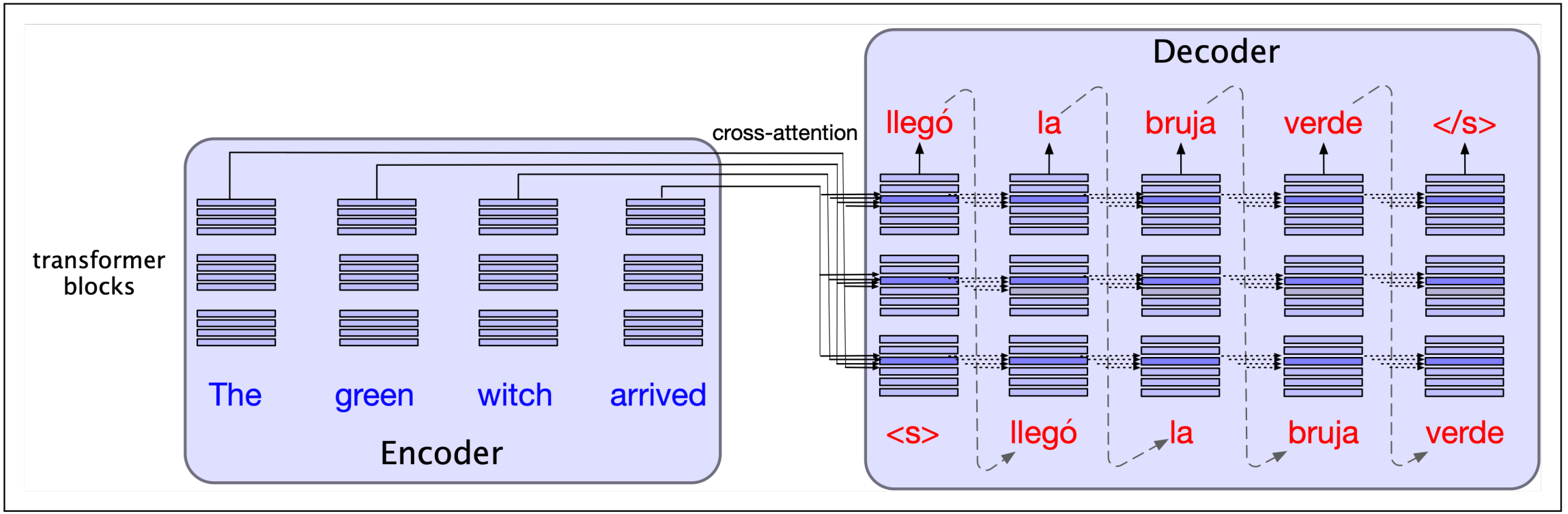


A sample alignment between sentences in English and French, with sentences extracted from Antoine de Saint-Exupery’s *Le Petit Prince* and a hypothetical translation. Sentence alignment takes sentences $e_1, ..., e_n$, and $f_1, ..., f_n$ and finds minimal sets of sentences that are translations of each other, including single sentence mappings like (e_1,f_1), (e_4,f_3), (e_5,f_4), (e_6,f_6) as well as 2-1 alignments (e_2/e_3,f_2), (e_7/e_8,f_7), and null alignments (f_5).

- Given two documents that are translations of each other, we generally need two steps to produce sentence alignments:
  - a cost function that takes a span of source sentences and a span of target sentences and returns a score measuring how likely these spans are to be translations.
  - an alignment algorithm that takes these scores to find a good alignment between the documents.
- To score the similarity of sentences across languages, we need to make use of a multilingual embedding space, in which sentences from different languages are in the same embedding space

$$c(x,y) = \frac{(1-\cos(x,y))\text{nSents}(x)\ \text{nSents}(y)}{\sum_{s=1}^{S} 1-\cos(x,y_s) + \sum_{s=1}^{S} 1-\cos(x_s,y)}$$

Details of the Encoder-Decoder Model



The encoder-decoder transformer architecture for machine translation. The encoder uses the transformer blocks, while the decoder uses a more powerful block with an extra cross-attention layer that can attend to all the encoder words.

- Encoder-decoder architecture is made up of two transformers:
  - an encoder, which is the same as the basic transformers that we have seen earlier,
  - and a decoder, which is augmented with a special new layer called the cross-attention layer.
- The encoder takes the source language input word tokens $X = x_1,...,x_n$ and maps them to an output representation $H^{enc} = h_1,...,h_n$; via a stack of encoder blocks.
- The decoder is essentially a conditional language model that attends to the encoder representation and generates the target words one by one

- Decoding can use any of the decoding methods like greedy, or temperature or nucleus sampling.
- But the most common decoding algorithm for MT is the beam search algorithm.
- In order to attend to the source language, the transformer blocks in the decoder have an extra cross-attention layer.
- The decoder transformer block includes an extra layer with a special kind of attention, cross-attention (also sometimes called encoder-decoder attention or source attention).

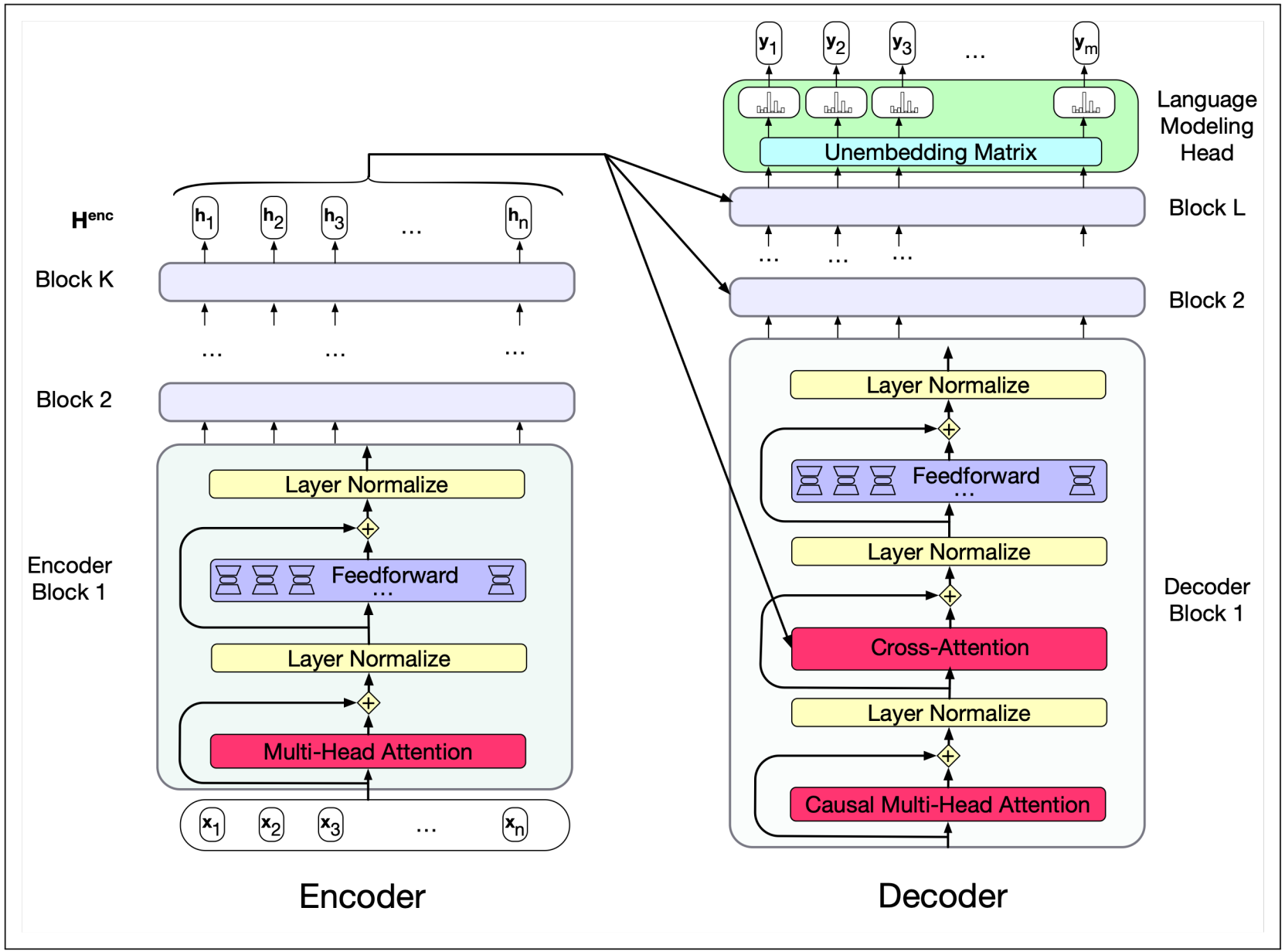
- Cross-attention has the same form as the multi-head attention in a normal transformer block, except that while the queries as usual come from the previous layer of the decoder, the keys and values come from the e output of the encoder.
- In standard multi-head attention the input to each attention layer is X, in cross attention the input is the the final output of the encoder $H^{enc} = h_1,...,h_n$. H^{enc} is of shape [n ×d], each row representing one input token.
- To link the keys and values from the encoder with the query from the prior layer of the decoder, we multiply the encoder output H^{enc} by the cross-attention layer's key weights W^K and value weights W^V.

- The query comes from the output from the prior decoder layer $H^{dec}[\ell-1]$, which is multiplied by the cross-attention layer's query weights W^Q

$$\mathbf{Q} = \mathbf{H}^{dec[\ell-1]}\mathbf{W}^{\mathbf{Q}};\ \mathbf{K} = \mathbf{H}^{enc}\mathbf{W}^{\mathbf{K}};\ \mathbf{V} = \mathbf{H}^{enc}\mathbf{W}^{\mathbf{V}}$$

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

- The cross attention thus allows the decoder to attend to each of the source language words as projected into the entire encoder final output representations.
- The other attention layer in each decoder block, the multi-head attention layer, is the same causal (left-to-right) attention that we saw earlier.
- The multi-head attention in the encoder, however, is allowed to look ahead at the entire source language text, so it is not masked.



The transformer block for the encoder and the decoder. The final output of the encoder $\mathbf{H}^{enc} = \mathbf{h}_1, ..., \mathbf{h}_n$ is the context used in the decoder. The decoder is a standard transformer except with one extra layer, the **cross-attention** layer, which takes that encoder output $\mathbf{H}^{enc}$ and uses it to form its **K** and **V** inputs.

- To train an encoder-decoder model, we use the same self-supervision model we used for training encoder-decoders RNNs
- The network is given the source text and then starting with the separator token is trained autoregressively to predict the next token using cross-entropy loss.
- cross-entropy loss for language modeling is determined by the probability the model assigns to the correct next word.
- So at time t the CE loss is the negative log probability the model assigns to the next word in the training sequence:

$$L_{CE}(\hat{\mathbf{y}}_t, \mathbf{y}_t) = -\log \hat{\mathbf{y}}_t[w_{t+1}]$$

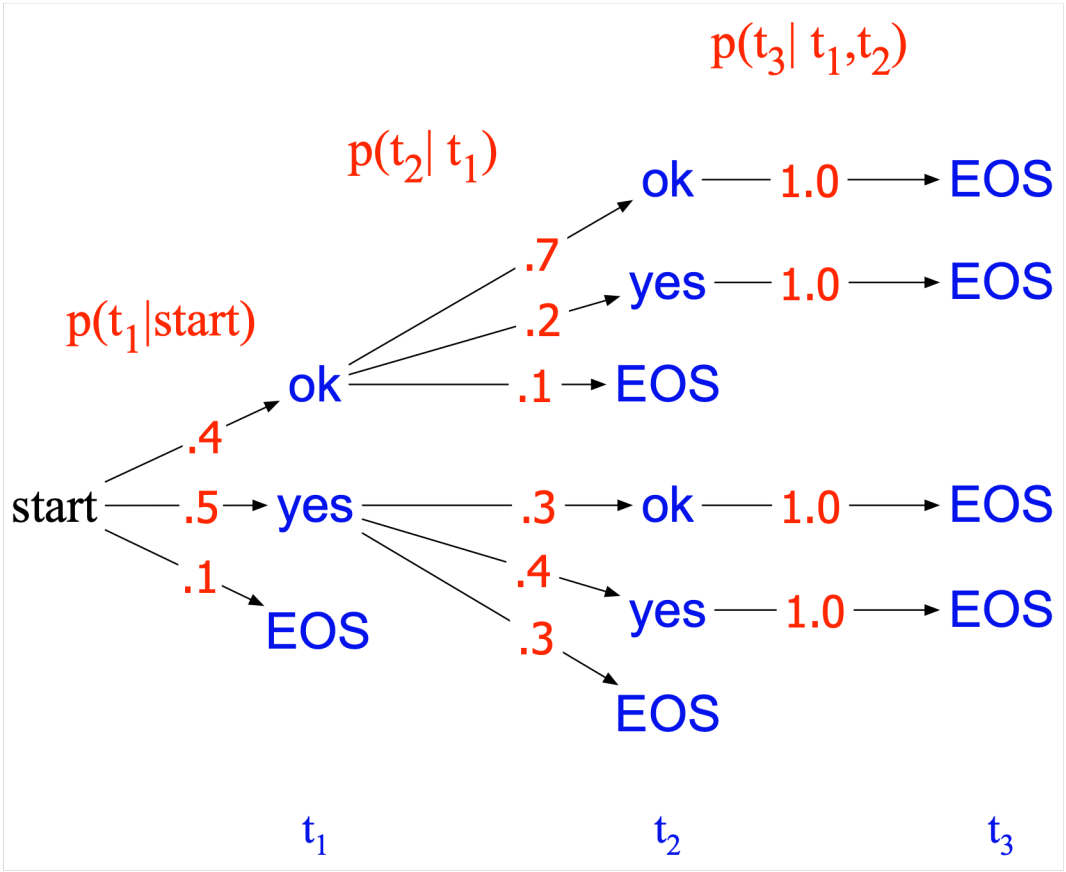
- As in that case, we use teacher forcing in the decoder. Recall that in teacher forcing, at each time step in decoding we force the system to use the gold target token from training as the next input x_{t+1}, rather than allowing it to rely on the (possibly erroneous) decoder output $\hat{y}_t$.

Decoding in MT: Beam Search

- In the greedy decoding algorithm, at each time step t in generation, the output y_t is chosen by computing the probability for each word in the vocabulary and then choosing the highest probability word (the argmax):

$$\hat{w}_t = \text{argmax}_{w \in V} P(w|\mathbf{w}_{<t})$$

- A problem with greedy decoding is that what looks high probability at word t might turn out to have been the wrong choice once we get to word t + 1.



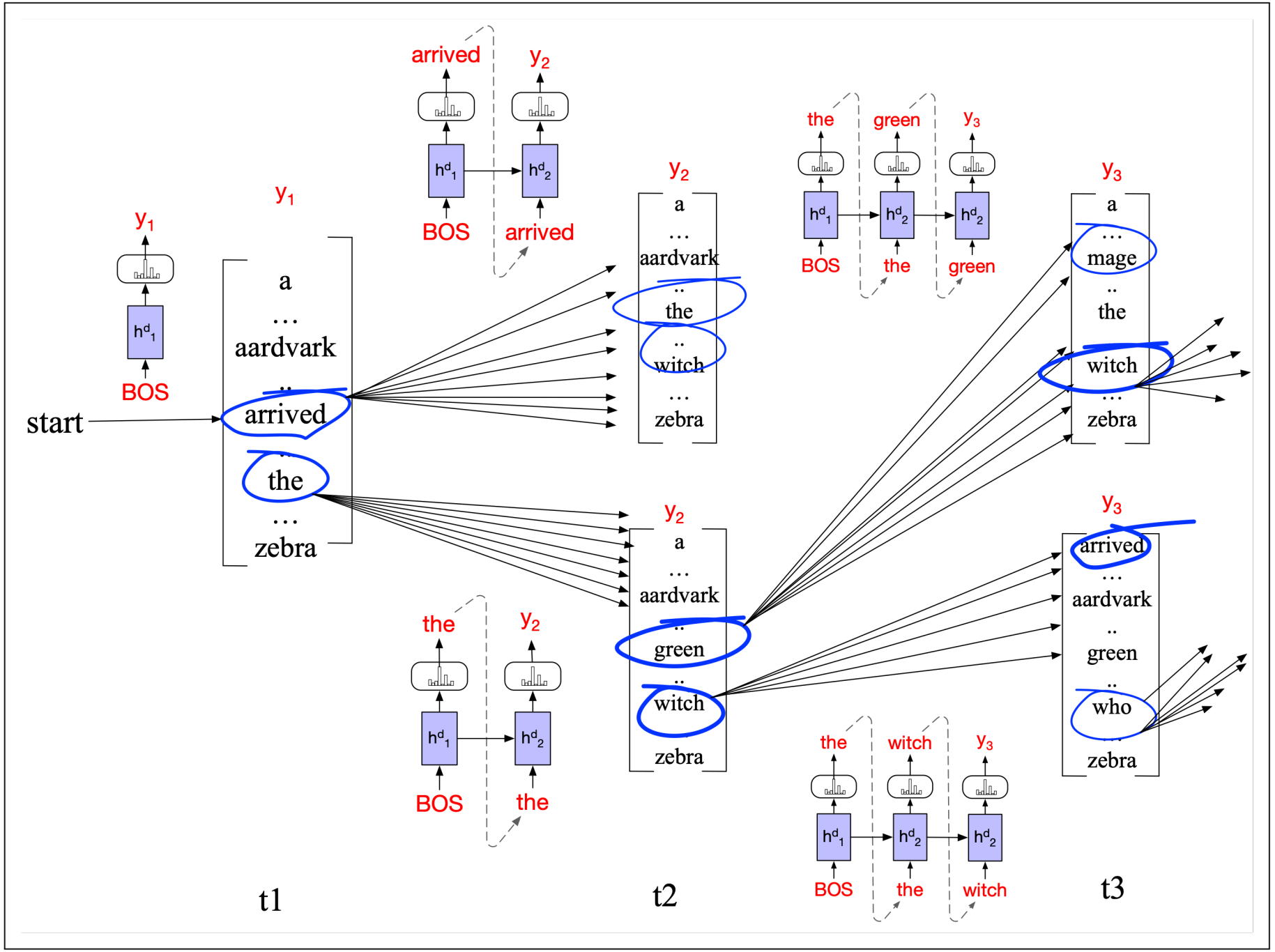
A search tree for generating the target string $T = t_1, t_2, \ldots$ from vocabulary $V = \{\text{yes}, \text{ok}, \texttt{<s>}\}$, showing the probability of generating each token from that state. Greedy search chooses *yes* followed by *yes*, instead of the globally most probable sequence *ok ok*.

- In beam search we model decoding as searching the space of possible generations, represented as a search tree whose branches represent actions (generating a token), and nodes represent states (having generated a particular prefix).
- We search for the best action sequence, i.e., the string with the highest probability.

Beam search

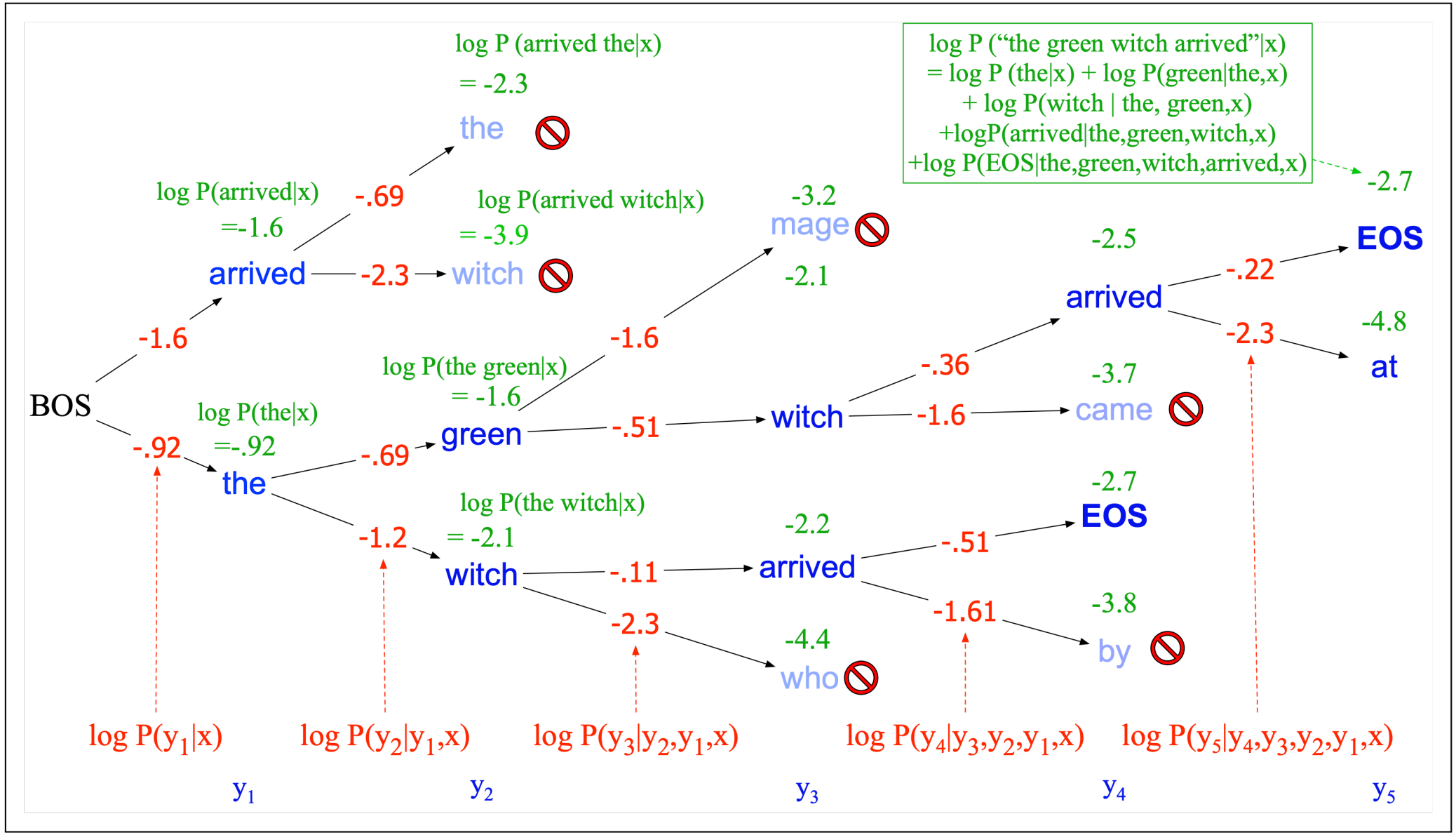
- MT systems generally decode using beam search, a heuristic search method first proposed by Lowerre (1976).
- In beam search, instead of choosing the best token to generate at each timestep, we keep k possible tokens at each step.
- This k is called the beam width
- At the first step of decoding, we compute a softmax over the entire vocabulary, assigning a probability to each word.
- We then select the k-best options from this softmax output.
- These initial k outputs are the search frontier and these k initial words are called hypotheses.

- A hypothesis is an output sequence, a translation-so-far, together with its probability.
- At subsequent steps, each of the k best hypotheses is extended incrementally by being passed to distinct decoders, which each generate a softmax over the entire vocabulary to extend the hypothesis to every possible next token.
- Each of these k×V hypotheses is scored by $P(y_i|x,y<i)$: the product of the probability of the current word choice multiplied by the probability of the path that led to it.
- We then prune the k ×V hypotheses down to the k best hypotheses, so there are never more than k hypotheses at the frontier of the search, and never more than k decoders.


start
y1
BOS
a
…
aardvark
..
arrived
..
the
…
zebra
y2
arrived
BOS
y2
a
…
aardvark
..
the
..
witch
…
zebra
y2
the
BOS
y2
a
…
aardvark
..
green
..
witch
zebra
y3
the
BOS
green
the
y3
witch
BOS
the
y3
a
…
mage
..
the
..
witch
…
zebra
y3
arrived
…
aardvark
..
green
..
who
…
zebra
t1
t2
t3

- This process continues until an EOS is generated indicating that a complete candidate output has been found.
- At this point, the completed hypothesis is removed from the frontier and the size of the beam is reduced by one.
- The search continues until the beam has been reduced to 0. The result will be k hypotheses.

$$
\begin{aligned}
score(y) &= \log P(y|x) \\
&= \log \left(P(y_1|x) P(y_2|y_1, x) P(y_3|y_1, y_2, x) \ldots P(y_t|y_1, \ldots, y_{t-1}, x) \right) \\
&= \sum_{i=1}^{t} \log P(y_i|y_1, \ldots, y_{i-1}, x)
\end{aligned}
$$



Scoring for beam search decoding with a beam width of $k = 2$. We maintain the log probability of each hypothesis in the beam by incrementally adding the logprob of generating each next token. Only the top k paths are extended to the next step.

Translating in low-resource situations: Data Augmentation

- Data augmentation is a statistical technique for dealing with insufficient training data, by adding new synthetic data that is generated from the current natural data.
- The most common data augmentation technique for machine translation is called backtranslation.
- Backtranslation relies on the intuition that while parallel corpora may be limited for particular languages or domains, we can often find a large (or at least larger) monolingual corpus, to add to the smaller parallel corpora that are available.
- The algorithm makes use of monolingual corpora in the target language by creating synthetic bitexts.

- In backtranslation, our goal is to improve source-to-target MT, given a small parallel text (a bitext) in the source/target languages and some monolingual data in the target language.
- We first use the bitext to train a MT system in the reverse direction: a target-to-source MT system.
- We then use it to translate the monolingual target data to the source language.
- Now we can add this synthetic bitext (natural
- target sentences, aligned with MT-produced source sentences) to our training data, and retrain our source-to-target MT model.
- In general backtranslation works surprisingly well; one estimate suggests that a system trained on backtranslated text gets about 2/3 of the gain as would training on the same amount of natural bitext

MT Evaluation

- Translations are evaluated along two dimensions:
  1. adequacy: how well the translation captures the exact meaning of the source sentence. Sometimes called faithfulness or fidelity.
  2. fluency: how fluent the translation is in the target language (is it grammatical, clear, readable, natural).
- Using humans to evaluate is most accurate, but automatic metrics are also used for convenience.

Using Human Raters to Evaluate MT

- The most accurate evaluations use human raters, such as online crowdworkers, to evaluate each translation along the two dimensions.
- For example, along the dimension of fluency, we can ask how intelligible, how clear, how readable, or how natural the MT output (the target text) is.
- We can give the raters a scale, for example, from 1 (totally unintelligible) to 5 (totally intelligible), or 1 to 100, and ask them to rate each sentence or paragraph of the MT output.
- We can do the same thing to judge the second dimension, adequacy, using raters to assign scores on a scale.

Automatic Evaluation

- Automatic Evaluation by Character Overlap: chrF
- The simplest and most robust metric for MT evaluation is called chrF, which stands for character F-score
- chrF is based on a simple intuition derived from the pioneering work of Miller and Beebe-Center (1956): a good machine translation will tend to contain characters and words that occur in a human translation of the same sentence.
- Consider a test set from a parallel corpus, in which each source sentence has both a gold human target translation and a candidate MT translation we'd like to evaluate.

- The chrF metric ranks each MT target sentence by a function of the number of character n-gram overlaps with the human translation.
- Given the hypothesis and the reference, chrF is given a parameter k indicating the length of character n-grams to be considered, and computes the average of the k precisions (unigram precision, bigram, and so on) and the average of the k recalls (unigram recall, bigram recall, etc.)
- chrP percentage of character 1-grams, 2-grams, ..., k-grams in the hypothesis that occur in the reference, averaged.
- chrR percentage of character 1-grams, 2-grams,..., k-grams in the reference that occur in the hypothesis, averaged.

- The metric then computes an F-score by combining chrP and chrR using a weighting parameter β . It is common to set β= 2, thus weighing recall twice as much as precision:

$$\mathrm{chrF}\beta = (1+\beta^2)\frac{\mathrm{chrP}\cdot\mathrm{chrR}}{\beta^2\cdot\mathrm{chrP}+\mathrm{chrR}}$$

For $\beta = 2$, that would be:

$$\mathrm{chrF2} = \frac{5\cdot\mathrm{chrP}\cdot\mathrm{chrR}}{4\cdot\mathrm{chrP}+\mathrm{chrR}}$$

- REF: witness for the past,
- HYP1: witness of the past, chrF2,2 = .86
- HYP2: past witness chrF2,2 = .62
- chrF ignores spaces, so we'll remove them from both the reference and hypothesis:
- REF: witnessforthepast,(18 unigrams, 17 bigrams)
- HYP1: witnessofthepast,(17 unigrams, 16 bigrams)
- unigrams that match: w i t n e s s f o t h e p a s t ,(17 unigrams)
- bigrams that match: wi it tn ne es ss th he ep pa as st t,(13 bigrams)

We use that to compute the unigram and bigram precisions and recalls:

unigram P:	17/17 = 1	unigram R:	17/18 = .944
bigram P:	13/16 = .813	bigram R:	13/17 = .765

Finally we average to get chrP and chrR, and compute the F-score:

$$\begin{aligned} \text{chrP} &= (17/17 + 13/16)/2 = .906 \\ \text{chrR} &= (17/18 + 13/17)/2 = .855 \\ \text{chrF2,2} &= 5\frac{\text{chrP} * \text{chrR}}{4\text{chrP} + \text{chrR}} = .86 \end{aligned}$$

Alternative overlap metric: BLEU

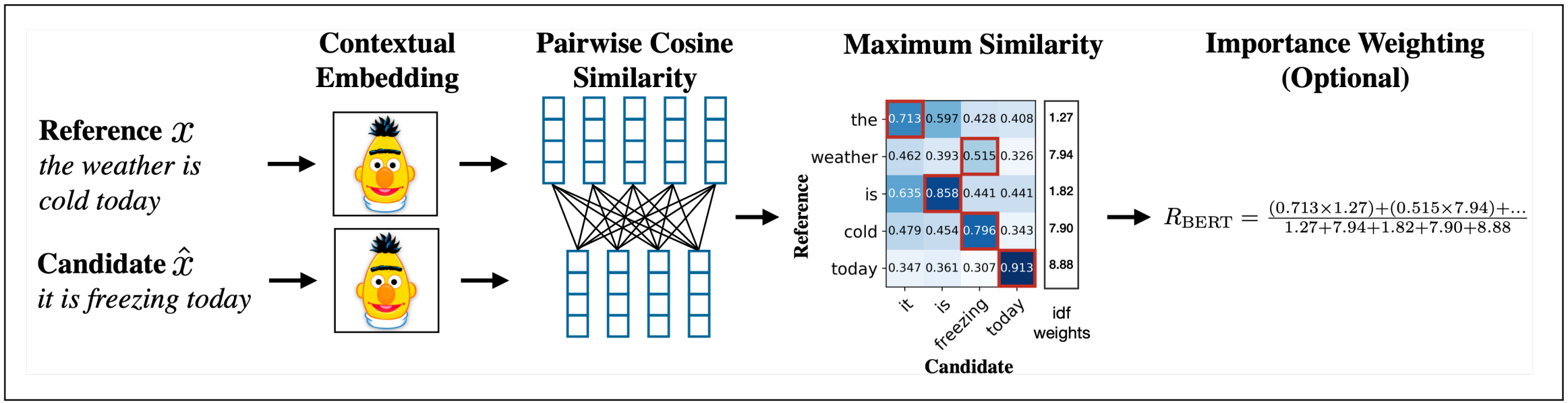
- There are various alternative overlap metrics.
- Before the development of chrF, it was common to use a word-based overlap metric called BLEU (for BiLingual Evaluation Understudy)
- It is purely precision-based rather than combining precision and recall
- The unigram precision for this corpus is the percentage of unigram tokens in the candidate translation that also occur in the reference translation, and ditto for bigrams and so on, up to 4-grams.
- BLEU extends this unigram metric to the whole corpus by computing the numerator as the sum over all sentences of the counts of all the unigram types that also occur in the reference translation, and the denominator is the total of the counts of all unigrams in all candidate sentences.
- We compute this n-gram precision for unigrams, bigrams, trigrams, and 4-grams and take the geometric mean.
- Because BLEU is a word-based metric, it is very sensitive to word tokenization, making it impossible to compare different systems if they rely on different tokenization standards, and doesn't work as well in languages with complex morphology.

Limitations

- While automatic character and word-overlap metrics like chrF or BLEU are useful, they have important limitations.
- chrF is very local: a large phrase that is moved around might barely change the chrF score at all

Automatic Evaluation: Embedding-Based Methods

- Scenario 1: we have human labelled dataset consists of tuples (x,$\tilde{x}$,r), where x = (x_1,...,x_n) is a reference translation, $\tilde{x}$ = ($\tilde{x}_1$,..., $\tilde{x}_m$) is a candidate machine translation, and r ∈R is a human rating that expresses the quality of $\tilde{x}$ with respect to x.
- Given such data, algorithms like COMET (Rei et al., 2020) BLEURT (Sellam et al., 2020) train a predictor on the human-labeled datasets which can finally predict r for a test set consists of (x,$\tilde{x}$).
- Scenario 2: We might not have such labelled dataset, in that case we can find embedding for x and $\tilde{x}$ and compare their embeddings.
- Each pair of tokens (x_i,$\tilde{x}_j$) is scored by its cosine $\frac{x_i . \tilde{x}_j}{|x_i|\,|\tilde{x}_j|}$


Contextual Embedding
Pairwise Cosine Similarity
Maximum Similarity
Importance Weighting (Optional)
Reference x
the weather is cold today
Candidate $\hat{x}$
it is freezing today
Reference
the 0.713 0.597 0.428 0.408 1.27
weather 0.462 0.393 0.515 0.326 7.94
is 0.635 0.858 0.441 0.441 1.82
cold 0.479 0.454 0.796 0.343 7.90
today 0.347 0.361 0.307 0.913 8.88
it
is
freezing
today
idf weights
Candidate
$R_{\text{BERT}} = \frac{(0.713\times1.27)+(0.515\times7.94)+\ldots}{1.27+7.94+1.82+7.90+8.88}$

Bias and Ethical Issues

- Machine translation raises many of the same ethical issues that we've discussed earlier.
- For example, consider MT systems translating from Hungarian (which has the gender neutral pronoun o") or Spanish (which often drops pronouns) into English (in which pronouns are obligatory, and they have grammatical gender).
- When translating a reference to a person described without specified gender, MT systems often default to male gender
- MT systems often assign gender according to culture stereotype.

- MT systems can be used in urgent situations where human translators may be unavailable or delayed:
  - in medical domains, to help translate when patients and doctors don't speak the same language,
  - or in legal domains, to help judges or lawyers communicate with witnesses or defendants.
- In order to 'do no harm', systems need ways to assign confidence values to candidate translations, so they can abstain from giving incorrect translations that may cause harm.