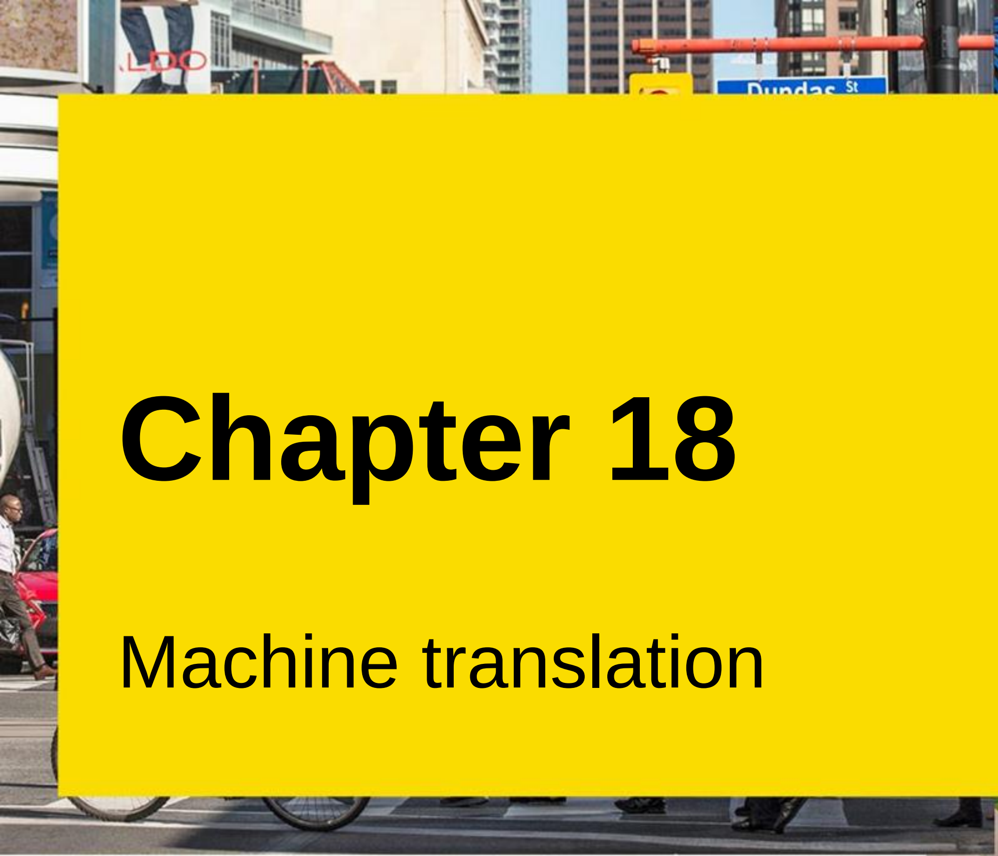


Chapter 18

Machine translation

Machine translation as a task

Machine translation can be formulated as an **optimization problem**

- $\boldsymbol{w}^{(s)}$ is a sentence in a **source language**
- $\boldsymbol{w}^{(t)}$ is a sentence in the **target language**
- Ψ is a **scoring function.**

$$\boxed{\hat{\boldsymbol{w}}^{(t)} = \underset{\boldsymbol{w}^{(t)}}{\operatorname{argmax}} \Psi(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)})}$$

This formalism requires **two components**:

1. A **decoding** algorithm for computing $\hat{\boldsymbol{w}}^{(t)}$
2. A **learning** algorithm for estimating the parameters of the scoring function Ψ

Ryerson University

Machine translation : Labeled Data Problem

- Labeled translation data usually comes in the form parallel sentences.

$$\boxed{\begin{aligned} \boldsymbol{w}^{(s)} &= \textit{A Vinay le gusta las manzanas.} \\ \boldsymbol{w}^{(t)} &= \textit{Vinay likes apples.} \end{aligned}}$$

- A useful feature function would note the translation pairs:

  (gusta, likes) (manzanas, apples) (Vinay, Vinay)

- **Problem:** Such word-to-word alignment is **not** given in the data.

Machine translation : Labeled Data **Solutions**

- **One solution** is to treat this alignment as a latent variable; this is the approach taken by classical statistical machine translation (SMT) systems.

- **Another solution** is to model the relationship between w(t) and w(s) through a complex and expressive function; like neural machine translation approach

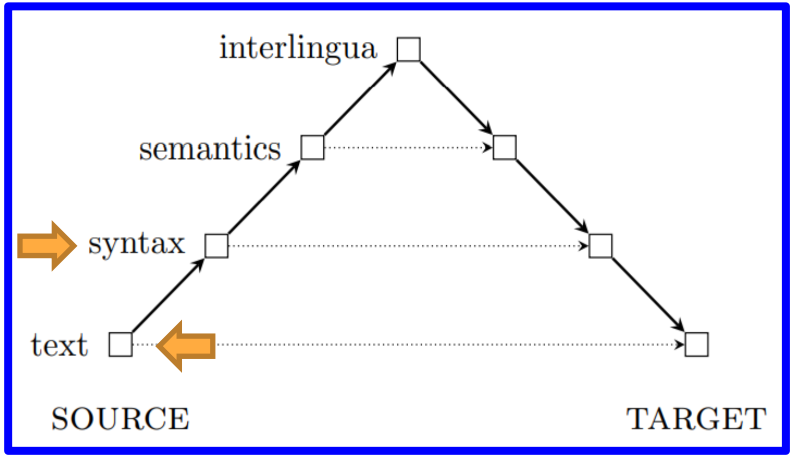
Ryerson University

Vauquois Pyramid

- The **Vauquois Pyramid** is a theory of how translation should be done.

- At the **lowest level**, the translation system operates on **individual words**, but the horizontal distance at this level is large, because languages express ideas differently.

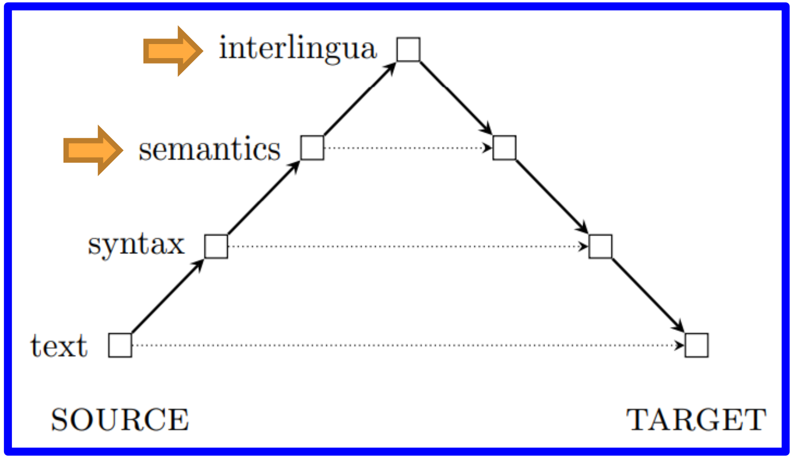
- By moving to **syntactic**, the translation distance is reduced; we then need only produce target-language text from the syntactic representation.



Vauquois Pyramid

- Further up the triangle lies **semantics**; translating between **semantic representations** should be easier still, but mapping between semantics and surface text is a difficult and unsolved problem

- At the **top of the triangle** is **interlingua**, a semantic representation that is so generic that it is identical across all human languages.



Ryerson University

Evaluating translations

There are **two main criteria** for a translation:

- **Adequacy**: The translation $\boldsymbol{w}^{(t)}$ should adequately reflect the **linguistic content** of $\boldsymbol{w}^{(s)}$

- **Fluency**: The translation $\boldsymbol{w}^{(t)}$ should read like fluent text in the target language.

Ryerson University

Example: Evaluating translations

$$w^{(s)} = \text{A Vinay le gusta Python}, \quad \left\{ \begin{aligned} w^{(t)} &= \text{Vinay likes Python} \\ w^{(t)}_{\text{gloss}} &= \text{To Vinay it like Python} \\ w^{(t)}_{3} &= \text{Vinay debugs memory leaks} \end{aligned} \right\}$$

- **Adequacy:**
  - $w^{(t)}_{\text{gloss}}$ or word-for-word translation is adequate because it has all the relevant content.
  - The $w^{(t)}_{3}$ is not adequate base on this criteria.

- **Fluency**: By this criterion, the gloss $w^{(t)}_{\text{gloss}}$ will score poorly, and $w^{(t)}_{3}$ will be preferred.

Ryerson University

Evaluating translations : BLEU Score

- Automated evaluations of machine translations typically merge **Adequacy** and **Fluency** criterias
- Evaluations performed by comparing the system translation with some reference translations, produced by human translators.

- The most popular quantitative metric is **BLEU**, which is based on n-gram precision.

- The n-gram precisions for three hypothesis translations

$$p_n = \frac{\text{number of } n\text{-grams appearing in both reference and hypothesis translations}}{\text{number of } n\text{-grams appearing in the hypothesis translation}}$$

[1

Ryerson University

BLEU and Brevity Penalty

- One potential weakness of **BLEU** is that it only measures precision
- The **BLEU** score is then based on the average → $\boxed{\exp \frac{1}{N} \sum_{n=1}^{N} \log p_n}$
- Two modifications are necessary for p_n
  - To avoid computing *log0*, all precisions are smoothed to ensure that they are positive
  - Each n-gram in the reference can be used at most once to avoid repetition.
  - Repetition : “to to to to to to” , Reference translation : “to be or not to be”

- Problem: precision-based metrics are biased in favor of short translations.
- Solution : To avoid this issue, a brevity penalty “BP” is applied to translations that are shorter than the reference

Ryerson University

BLEU and Brevity Penalty Example

- A reference translation and three system outputs.
- For each output p_n , the precision at each n-gram.
- BP indicates the brevity penalty

	Translation	p_1	p_2	p_3	p_4	BP	BLEU
Reference	*Vinay likes programming in Python*						
Sys1	*To Vinay it like to program Python*	$\frac{2}{7}$	0	0	0	1	.21
Sys2	*Vinay likes Python*	$\frac{3}{3}$	$\frac{1}{2}$	0	0	.51	.33
Sys3	*Vinay likes programming in his pajamas*	$\frac{4}{6}$	$\frac{3}{5}$	$\frac{2}{4}$	$\frac{1}{3}$	1	.76

Ryerson University

Evaluating translations : BLEU Score

- Automated metrics like **BLEU** have been validated by **correlation** with human judgments of translation quality.
- Nonetheless, it is not difficult to construct examples in which the **BLEU score is high**, yet the **translation is disfluent or carries a completely different meaning from the original**.
- Despite the importance of pronouns for **semantics**, they have a **marginal** impact on BLEU

Ryerson University

Pronouns Translation.

- Consider the problem of translating pronouns. Because pronouns refer to specific entities, a single incorrect pronoun can obliterate the **semantics** of the original sentence.

- The problem of **pronoun translation** intersects with issues of **fairness** and **bias**
    - In many languages, the third person singular pronoun is gender neutral.
    - This bias was not directly programmed into the translation model; it arises from statistical tendencies in existing datasets
    - If a dataset has even a slight tendency towards men as doctors, the resulting translation model may produce translations in which doctors are always he, and nurses are always she

Ryerson University

Other Translation metrics

- **METEOR** is a weighted F -MEASURE, which is a combination of recall and precision

- **Translation Error Rate (TER)** computes the string edit distance between the reference and the hypothesis

- **RIBES** metric applies rank correlation to measure the similarity in word order between the system and reference translations

Data

- Data-driven approaches to machine translation rely primarily on **parallel corpora**, which are translations at the **sentence level**.

- Many languages have sizable parallel corpora with some high-resource language, but not with each other.

- The high-resource language can then be used as a “pivot” or “bridge”
    - For example, use **Spanish** as a bridge for translation between **Catalan** and **English**.

- For most of the 6000 languages spoken today, the only source of translation data remains the **Judeo-Christian Bible**.
    - While relatively small, at less than a million tokens, the Bible has been translated into more than **2000 languages**

Ryerson University

Statistical machine translation : intro

- Review : Two main criteria for machine translation fluency and adequacy
- A **natural modeling** approach is to represent them with separate scores

$$\boxed{\Psi(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}) = \Psi_A(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}) + \Psi_F(\boldsymbol{w}^{(t)})}$$

- The fluency score Ψ_F need not even consider the source sentence; it only judges $\boldsymbol{w}^{(t)}$ on whether it is fluent in the target language.
  - This decomposition is advantageous because it makes it possible to estimate the two scoring functions on separate data.
- The fluency model can be estimated from monolingual text in the target language.
  - Large monolingual corpora are now available in many languages, thanks to resources such as Wikipedia.

Noisy channel model

- In the noisy channel model, each **scoring function** is a **log probability**

$$
\begin{aligned}
\Psi_A(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}) &\triangleq \log \mathrm{p}_{S|T}(\boldsymbol{w}^{(s)} \mid \boldsymbol{w}^{(t)}) \\
\Psi_F(\boldsymbol{w}^{(t)}) &\triangleq \log \mathrm{p}_T(\boldsymbol{w}^{(t)}) \\
\Psi(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}) &= \underbrace{\log \mathrm{p}_{S|T}(\boldsymbol{w}^{(s)} \mid \boldsymbol{w}^{(t)}) + \log \mathrm{p}_T(\boldsymbol{w}^{(t)})}_{\log \mathrm{p}_{S,T}} = \log \mathrm{p}_{S,T}(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)})
\end{aligned}
$$

- By setting the scoring functions equal to logarithms of the prior and likelihood, their sum is equal to $\log \mathrm{p}_{S,T}$, which is **logarithm of joint probability of source and target**.

- The sentence $\hat{\boldsymbol{w}}^{(t)}$ that maximizes this joint probability is also the maximizer of the conditional probability $\mathrm{p}_{T|S}$, making it the most likely target language sentence, conditioned on the source.

Statistical translation modeling : Alignment

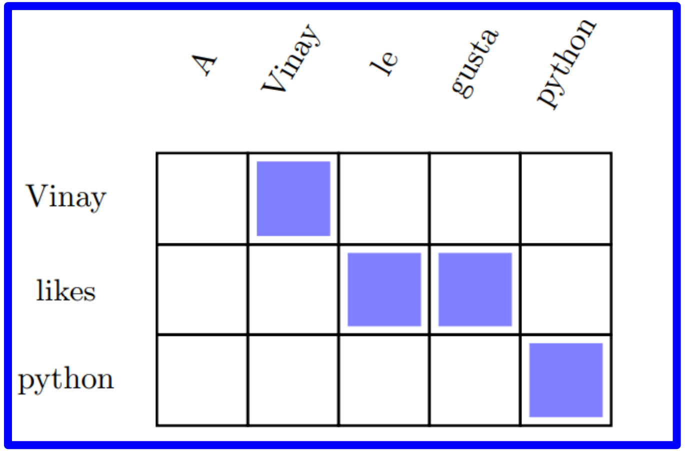
- The simplest decomposition of the translation model is **word-to-word**
- Each word in the **source** should be aligned to a word in the **translation**.
- This approach presupposes an **alignment** $\mathcal{A}(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)})$ which contains a **list of pairs** of **source** and **target** tokens.
- Each alignment contains exactly one tuple for each word in the source
- If no appropriate word in the target can be identified for a source word, it is aligned to ∅.
- Words in the target can align with multiple words in the source.

Ryerson University

Alignment Example

Example:

$$\left\{\begin{aligned} \boldsymbol{w}^{(s)} &= \textit{A Vinay le gusta Python} \\ \boldsymbol{w}^{(t)} &= \textit{Vinay likes Python} \end{aligned}\right.$$



$$\left\{\begin{aligned} \mathcal{A}(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}) &= \{(A, \varnothing), (\textit{Vinay, Vinay}), (\textit{le, likes}), (\textit{gusta, likes}), (\textit{Python,Python})\} \\ \mathcal{A}(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}) &= \{(\textit{A, Vinay}), (\textit{Vinay, likes}), (\textit{le, Python}), (\textit{gusta}, \varnothing), (\textit{Python}, \varnothing)\} \end{aligned}\right.$$

Ryerson University

Alignment and Translation

- The **joint probability** of the **alignment** and the **translation**:

$$\begin{aligned} \mathrm{p}(\boldsymbol{w}^{(s)}, \mathcal{A} \mid \boldsymbol{w}^{(t)}) &= \prod_{m=1}^{M^{(s)}} \mathrm{p}(w_m^{(s)}, a_m \mid w_{a_m}^{(t)}, m, M^{(s)}, M^{(t)}) \\ &= \prod_{m=1}^{M^{(s)}} \mathrm{p}(a_m \mid m, M^{(s)}, M^{(t)}) \times \mathrm{p}(w_m^{(s)} \mid w_{a_m}^{(t)}) \end{aligned}$$

Ryerson University

Statistical translation modeling

This probability model makes two key assumptions:

$$\mathrm{p}(\boldsymbol{w}^{(s)}, \mathcal{A} \mid \boldsymbol{w}^{(t)}) = \prod_{m=1}^{M^{(s)}} \mathrm{p}(w_m^{(s)}, a_m \mid w_{a_m}^{(t)}, m, M^{(s)}, M^{(t)})$$
$$= \prod_{m=1}^{M^{(s)}} \mathrm{p}(a_m \mid m, M^{(s)}, M^{(t)}) \times \mathrm{p}(w_m^{(s)} \mid w_{a_m}^{(t)})$$

- The alignment probability factors across tokens
- This means that each alignment decision is independent of the others, and depends only on the index m, and the sentence lengths $M^{(s)}$ and $M^{(t)}$

$$\mathrm{p}(\mathcal{A} \mid \boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}) = \prod_{m=1}^{M^{(s)}} \mathrm{p}(a_m \mid m, M^{(s)}, M^{(t)})$$

- The translation probability also factors across tokens
- so that each word in w(s) depends only on its aligned word in w(t) . This means that translation is word-to-word, ignoring context.

$$\mathrm{p}(\boldsymbol{w}^{(s)} \mid \boldsymbol{w}^{(t)}, \mathcal{A}) = \prod_{m=1}^{M^{(s)}} \mathrm{p}(w_m^{(s)} \mid w_{a_m}^{(t)})$$

Ryerson University

Statistical translation modeling

- To translate with such a model, we could sum or max over all possible alignments
- The term $\mathrm{p}(\mathcal{A})$ defines the prior probability over alignments.

$$\boxed{\begin{aligned}
\mathrm{p}(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}) =& \sum_{\mathcal{A}} \mathrm{p}(\boldsymbol{w}^{(s)}, \boldsymbol{w}^{(t)}, \mathcal{A}) \\
=& \mathrm{p}(\boldsymbol{w}^{(t)}) \sum_{\mathcal{A}} \mathrm{p}(\mathcal{A}) \times \mathrm{p}(\boldsymbol{w}^{(s)} \mid \boldsymbol{w}^{(t)}, \mathcal{A}) \\
\geq& \mathrm{p}(\boldsymbol{w}^{(t)}) \max_{\mathcal{A}} \mathrm{p}(\mathcal{A}) \times \mathrm{p}(\boldsymbol{w}^{(s)} \mid \boldsymbol{w}^{(t)}, \mathcal{A})
\end{aligned}}$$

Neural machine translation

- Neural network models for machine translation are based on the encoder-decoder architecture
- The **encoder** network converts the source language sentence into a vector or matrix representation
- The **decoder** network then converts the encoding into a sentence in the target language

$$\boxed{\begin{aligned} \boldsymbol{z} =& \textsc{Encode}(\boldsymbol{w}^{(s)}) \\ \boldsymbol{w}^{(t)} \mid \boldsymbol{w}^{(s)} \sim& \textsc{Decode}(\boldsymbol{z}), \end{aligned}}$$

Where the second line means that the function $\textsc{Decode}(\boldsymbol{z})$ defines the conditional probability $\mathrm{p}(\boldsymbol{w}^{(t)} \mid \boldsymbol{w}^{(s)})$

Ryerson University

Neural machine translation

- The **decoder** is typically a recurrent neural network, which generates the **target** language sentence one word at a time, while recurrently updating a hidden state.

- The encoder and decoder networks are trained **end-to-end** from **parallel sentences**.

- If the **output** layer of the decoder is a **logistic** function, then the entire architecture can be trained to maximize the conditional log-likelihood:

$$\log \mathrm{p}(\boldsymbol{w}^{(t)} \mid \boldsymbol{w}^{(s)}) = \sum_{m=1}^{M^{(t)}} \mathrm{p}(w_m^{(t)} \mid \boldsymbol{w}_{1:m-1}^{(t)}, \boldsymbol{z})$$

$$\mathrm{p}(w_m^{(t)} \mid \boldsymbol{w}_{1:m-1}^{(t)}, \boldsymbol{w}^{(s)}) \propto \exp\left(\boldsymbol{\beta}_{w_m^{(t)}} \cdot \boldsymbol{h}_{m-1}^{(t)}\right)$$

Ryerson University

Neural machine translation

- When the **output** layer of decoder is **logistic** function:

$$\log p(\boldsymbol{w}^{(t)} \mid \boldsymbol{w}^{(s)}) = \sum_{m=1}^{M^{(t)}} p(w_m^{(t)} \mid \boldsymbol{w}_{1:m-1}^{(t)}, \boldsymbol{z})$$

$$p(w_m^{(t)} \mid \boldsymbol{w}_{1:m-1}^{(t)}, \boldsymbol{w}^{(s)}) \propto \exp\left(\boldsymbol{\beta}_{w_m^{(t)}} \cdot \boldsymbol{h}_{m-1}^{(t)}\right)$$

- Where the hidden state $\boldsymbol{h}_{m-1}^{(t)}$ is a **recurrent function** of the previously generated text $\boldsymbol{w}_{1:m-1}^{(t)}$ and the encoding $\boldsymbol{z}$.

$$\boxed{p(w_m^{(t)} \mid \boldsymbol{w}_{1:m-1}^{(t)}, \boldsymbol{w}^{(s)}) \propto \exp\left(\boldsymbol{\beta}_{w_m^{(t)}} \cdot \boldsymbol{h}_{m-1}^{(t)}\right)} = \boxed{w_m^{(t)} \mid \boldsymbol{w}_{1:m-1}^{(t)}, \boldsymbol{w}^{(s)} \sim \mathrm{SoftMax}\left(\boldsymbol{\beta} \cdot \boldsymbol{h}_{m-1}^{(t)}\right)}$$

- Where $\boldsymbol{\beta} \in \mathbb{R}^{(V^{(t)} \times K)}$ is the **matrix of output word vectors** for the $V^{(t)}$ words in the target language vocabulary.

Ryerson University

Sequence-to-Sequence Model

- The **simplest** encoder-decoder architecture is the sequence-to-sequence model
- Sequence-to-sequence translation is nothing more than wiring together two LSTMs: one to **read the source**, and another to **generate the target**.
- In this model, the encoder is set to the final hidden state of a long short-term memory (LSTM) on the source sentence

$$\boxed{\begin{aligned} \boldsymbol{h}_m^{(s)} &= \mathrm{LSTM}(\boldsymbol{x}_m^{(s)}, \boldsymbol{h}_{m-1}^{(s)}) \\ \boldsymbol{z} &\triangleq \boldsymbol{h}_{M^{(s)}}^{(s)}, \end{aligned}}$$

- Where $\boldsymbol{x}_m^{(s)}$ is the embedding of source language word $w_m^{(s)}$.
- The encoding then provides the initial hidden state for the decoder LSTM

$$\boxed{\begin{aligned} \boldsymbol{h}_0^{(t)} &= \boldsymbol{z} \\ \boldsymbol{h}_m^{(t)} &= \mathrm{LSTM}(\boldsymbol{x}_m^{(t)}, \boldsymbol{h}_{m-1}^{(t)}), \end{aligned}}$$

Tweaks on Sequence-to-Sequence Model

- The model works better if the source sentence is **reversed**, reading from the end of the sentence back to the beginning.

- In this way, the words at the **beginning** of the **source** have the **greatest** impact on the **encoding z**, and therefore impact the words at the **beginning** of the **target** sentence.

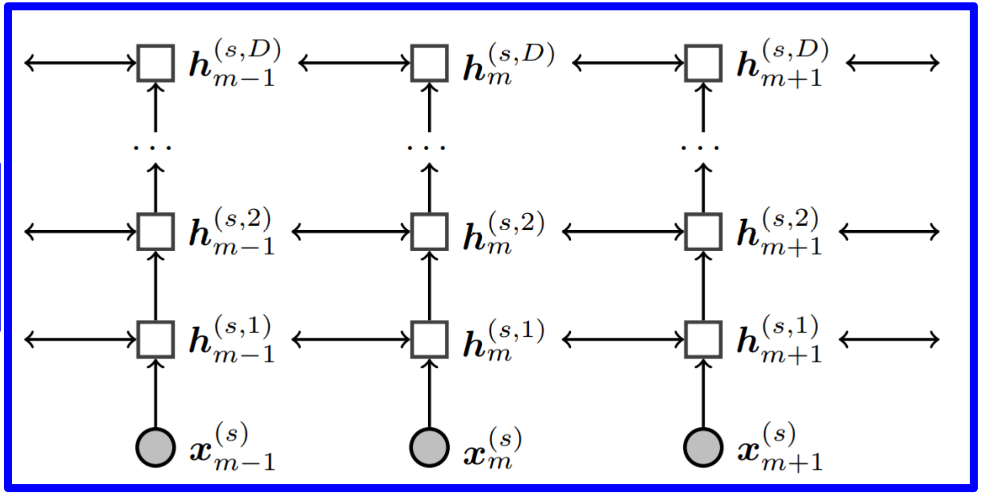
- Advance encoding models, such as **neural attention** has **eliminated** the need for **reversing** the source sentence.

Ryerson University

Tweaks on Sequence-to-Sequence Model : pt2

- The encoder and decoder can be implemented as **deep LSTMs**, with multiple layers of hidden states
- Each hidden state $\boldsymbol{h}_m^{(s,i)}$ at layer ***i*** is treated as input to an LSTM at layer ***i* + *1***

$$
\begin{aligned}
\boldsymbol{h}_m^{(s,1)} &= \mathrm{LSTM}(\boldsymbol{x}_m^{(s)}, \boldsymbol{h}_{m-1}^{(s)}) \\
\boldsymbol{h}_m^{(s,i+1)} &= \mathrm{LSTM}(\boldsymbol{h}_m^{(s,i)}, \boldsymbol{h}_{m-1}^{(s,i+1)}), \quad \forall i \geq 1
\end{aligned}
$$



Ryerson University

Tweaks on Sequence-to-Sequence Model : pt3

- Significant improvements can be obtained by creating an ensemble of translation models, each trained from a different random initialization.

- For an ensemble of size **N**, the per-token decoding probability is set equal to:

$$\boxed{\mathrm{p}(w^{(t)} \mid \boldsymbol{z}, \boldsymbol{w}_{1:m-1}^{(t)}) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{p}_i(w^{(t)} \mid \boldsymbol{z}, \boldsymbol{w}_{1:m-1}^{(t)})}$$

- where p_i is the decoding probability for model ***i***

Ryerson University

Model comparison

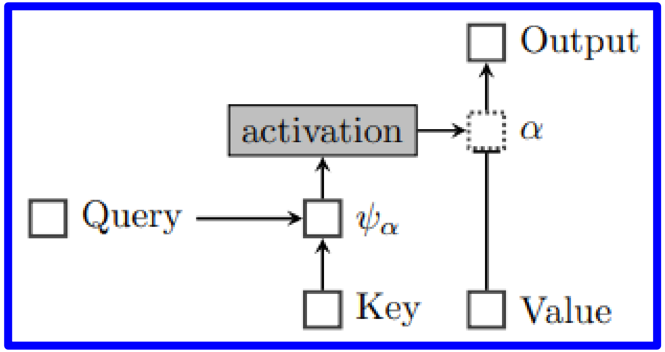
- Statistical Translation approach is **compositional**
- Encoder-decoder models are **contextualized**

- Question: Is it possible for translation to be both **contextualized** and **compositional**?
- Answer: Yes, One approach is to augment neural translation with an **attention mechanism**.

Ryerson University

Neural Attention

- **Neural attention** makes it possible to integrate **alignment** into the **encoder-decoder architecture.**
- In general, attention can be thought of as using a query to select from a memory of **key-value pairs**.
- However, the query, keys, and values are all vectors, and the entire operation is differentiable
- A general view of neural attention:

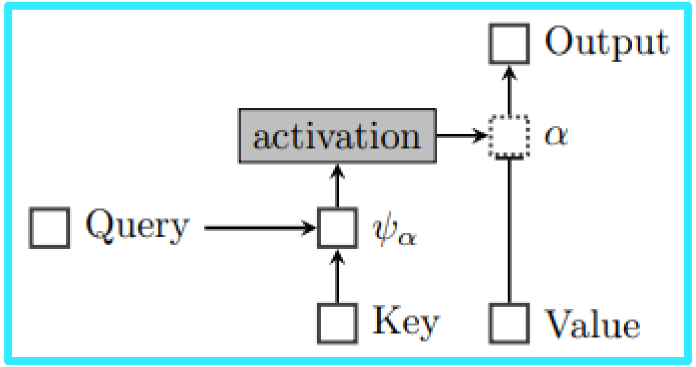


Ryerson University

Neural Attention

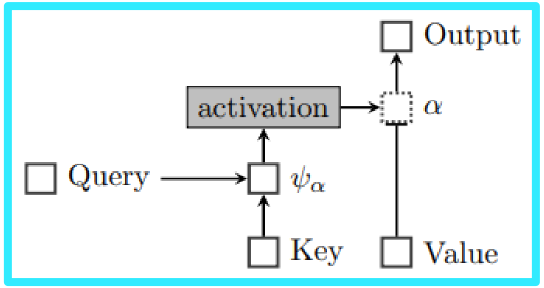
- For each key n in the memory, score $\psi_\alpha(m, n)$ gets computed with respect to the query m
- $\psi_\alpha(m, n)$ score is a function of the **compatibility** of **key** and **query**
- $\psi_\alpha(m, n)$ can be computed using a **small feedforward neural network**.
- The **vector of scores** is passed through an **activation function**, like SoftMax.
- Output of activation function is a vector of non-negative numbers:

$$\boxed{[\alpha_{m\to1}, \alpha_{m\to2}, \ldots, \alpha_{m\to N}]^\top}$$



Ryerson University

Neural Attention



- Output of activati n function is a vector of non-negative numbers $[\alpha_{m\to 1}, \alpha_{m\to 2}, \ldots, \alpha_{m\to N}]^\top$
- $[\alpha_{m\to 1}, \alpha_{m\to 2}, \ldots, \alpha_{m\to N}]^\top$ has length of N, equal to the size of the memory.
- Each value in the memory $\boldsymbol{v}_n$ is multiplied by the attention $\alpha_{m\to n}$; the **sum** of these scaled values is the output
- The dotted box indicates that each $\alpha_{m\to n}$ can be viewed as a **gate** on value n
- Extreme case: $\alpha_{m\to n} = 1$ and $\alpha_{m\to n'} = 0$ for all other n'
  Then the attention mechanism selects the value $\boldsymbol{v}_n$ from memory.

Ryerson University

Neural Attention : Decoding

- At each step m in decoding, attentional state is computed by executing a query, which is equal to the **state of decoder**, $\boldsymbol{h}_m^{(t)}$. The resulting compatibility scores :

$$\boxed{\psi_\alpha(m,n) = \boldsymbol{v}_\alpha \cdot \tanh(\Theta_\alpha[\boldsymbol{h}_m^{(t)};\boldsymbol{h}_n^{(s)}])}$$

- The function ψ is thus a **two-layer feedforward neural network**, with weights $\boldsymbol{v}_\alpha$ on the **output layer**, and weights $\boldsymbol{\Theta}_\alpha$ on the **input layer**.

- To convert these **scores** into **attention weights**, we apply an activation function, which can be vector-wise SoftMax or an element-wise sigmoid

Neural Attention : Activation functions

- Vector-wise SoftMax attention:

$$\alpha_{m\to n} = \frac{\exp \psi_\alpha(m,n)}{\sum_{n'=1}^{M^{(s)}} \exp \psi_\alpha(m,n')}$$

- Element-wise sigmoid attention:

$$\alpha_{m\to n} = \sigma\left(\psi_\alpha(m,n)\right)$$

$$\boldsymbol{c}_m = \sum_{n=1}^{M^{(s)}} \alpha_{m\to n}\boldsymbol{z}_n$$

$$\tilde{\boldsymbol{h}}_m^{(t)} = \tanh\left(\Theta_c[\boldsymbol{h}_m^{(t)}; \boldsymbol{c}_m]\right)$$

$$\mathrm{p}(w_{m+1}^{(t)} \mid \boldsymbol{w}_{1:m}^{(t)}, \boldsymbol{w}^{(s)}) \propto \exp\left(\boldsymbol{\beta}_{w_{m+1}^{(t)}} \cdot \tilde{\boldsymbol{h}}_m^{(t)}\right)$$

  - Attention α is then used to compute a context vector c_m by taking a weighted average over the columns of Z.
  - $\alpha_{m\to n} \in [0,1]$ is the **amount of attention** from **word** m of the **target** to word n of the **source**.
  - The decoder state $\boldsymbol{h}_m^{(t)}$ is concatenated with the context vector, forming the input to compute a final output $\tilde{\boldsymbol{h}}_m^{(t)}$