

SPRINT 24+

Attention



このスライドは？

ここでは、Attentionの
基本的な知識を学びましょう

Attentionとは



<https://arxiv.org/pdf/1502.03044.pdf>

人間は、関心のある対象がシーンの特定の部分に現れることに気づくと、将来、シーンの部分であるその領域に注意（Attention）を集中する傾向があると考えられています。

Attentionメカニズムの起源は、シーン全体を把握する代わりに、選択的に対象へ注意を向ける人間の知覚メカニズムに由来しています。

このアルゴリズムは、1990年代よりコンピュータビジョンの領域で用いられていました。

2014年に発表された論文のなかで、RNNを用いた画像分類タスクにAttentionアルゴリズムが適用されたことから注目を集め[1]、さらに、Dzmitry Bahdanau, et al (2014)においてニューラルネットワークによる機械翻訳タスクで長い入力文から選択的に情報を引き出すことに応用されました[2]。

こうした経緯を経て、自然言語処理の領域にAttentionアルゴリズムが導入されました。

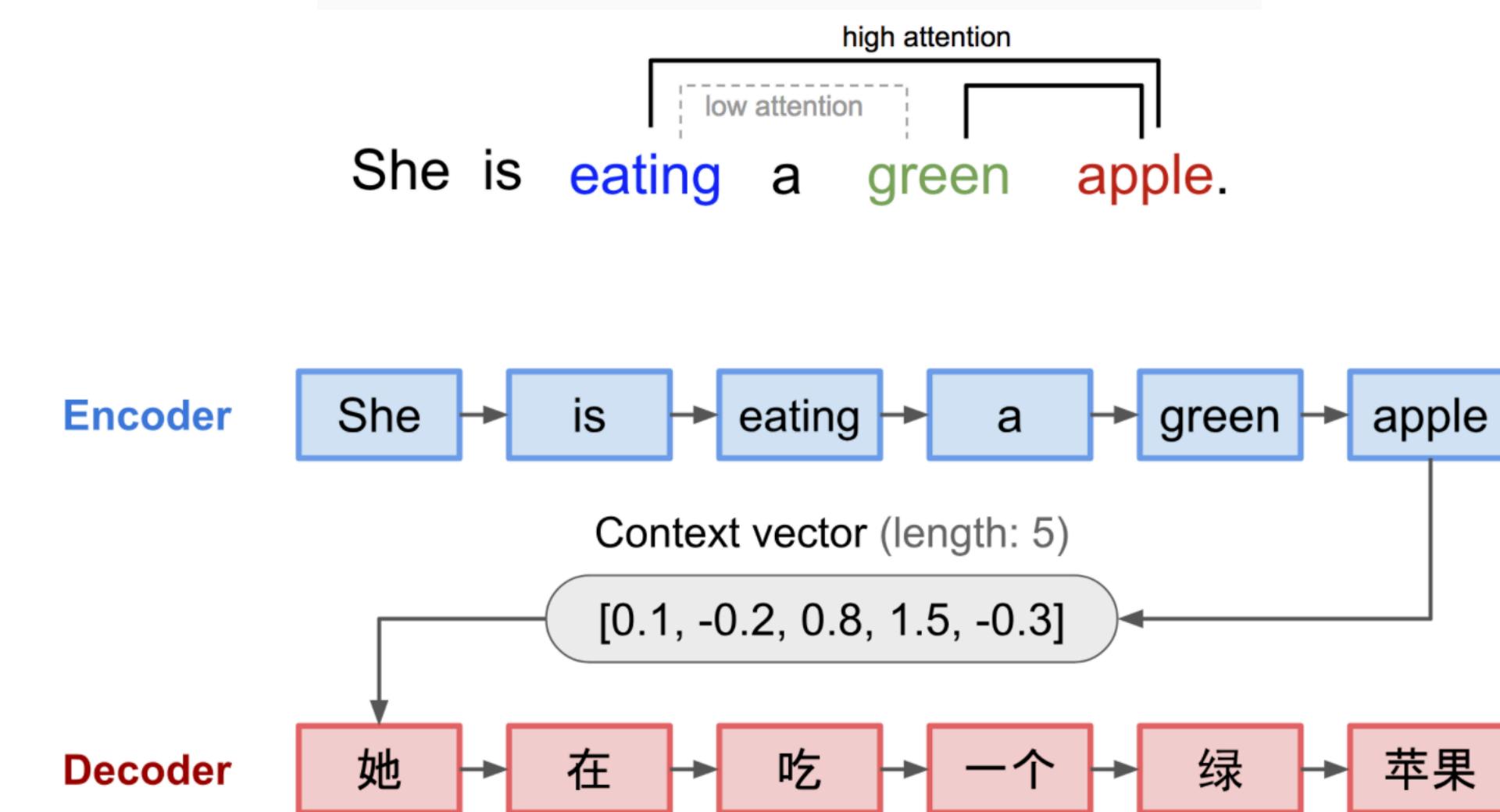
[1]Volodymyr Mnih, Nicolas Heess, Alex Graves and Koray Kavukcuoglu. Recurrent Models of Visual Attention. arXiv preprint arXiv:1406.6247v1, 2014

[2]Dzmitry Bahdanau, KyungHyun Cho and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv preprint arXiv:1409.0473v7, 2014

なぜAttentionが必要なのか

seq2seqを用いた機械翻訳タスクにおける課題は、エンコーダからデコーダへ渡される**コンテキストベクトル**（いわゆる長期記憶）が、**長い入力文の情報を保持することの難しさ**にありました。

多くの場合、入力全体の処理が完了すると、**最初の部分を忘れてしまします**。この問題を解決するために、Attentionアルゴリズムが採用されました。



Attentionとは何か

Attentionとは、ネットワークアーキテクチャの一つの機構であり、そのアルゴリズムの核心は、デコーダから出力される各単語に対して、エンコーダの全出力への依存をマッピングし、どれだけ依存しているかの度合いを定量化することです。

このマッピングはコンテキストベクトルによって保持されるため、**入力全体を忘れるという問題を回避**することができます。

Attentionは何をしている？

こうした依存の度合いを求めるために、Attention機構では
以下のような手続きを行います。

デコーダの各タイムステップにおいて、その時点でのデ
コーダの隠れ状態に関して、エンコーダの各隠れ状態との
アライメントスコア（配列の類似性を評価するマッチング
スコア）を計算します。

アライメントスコアは、次のデコーダ出力を生成すると
きに**デコーダが各エンコーダ出力に配る注意**

(Attention) の度合いを定量化したものと解釈されま
す。

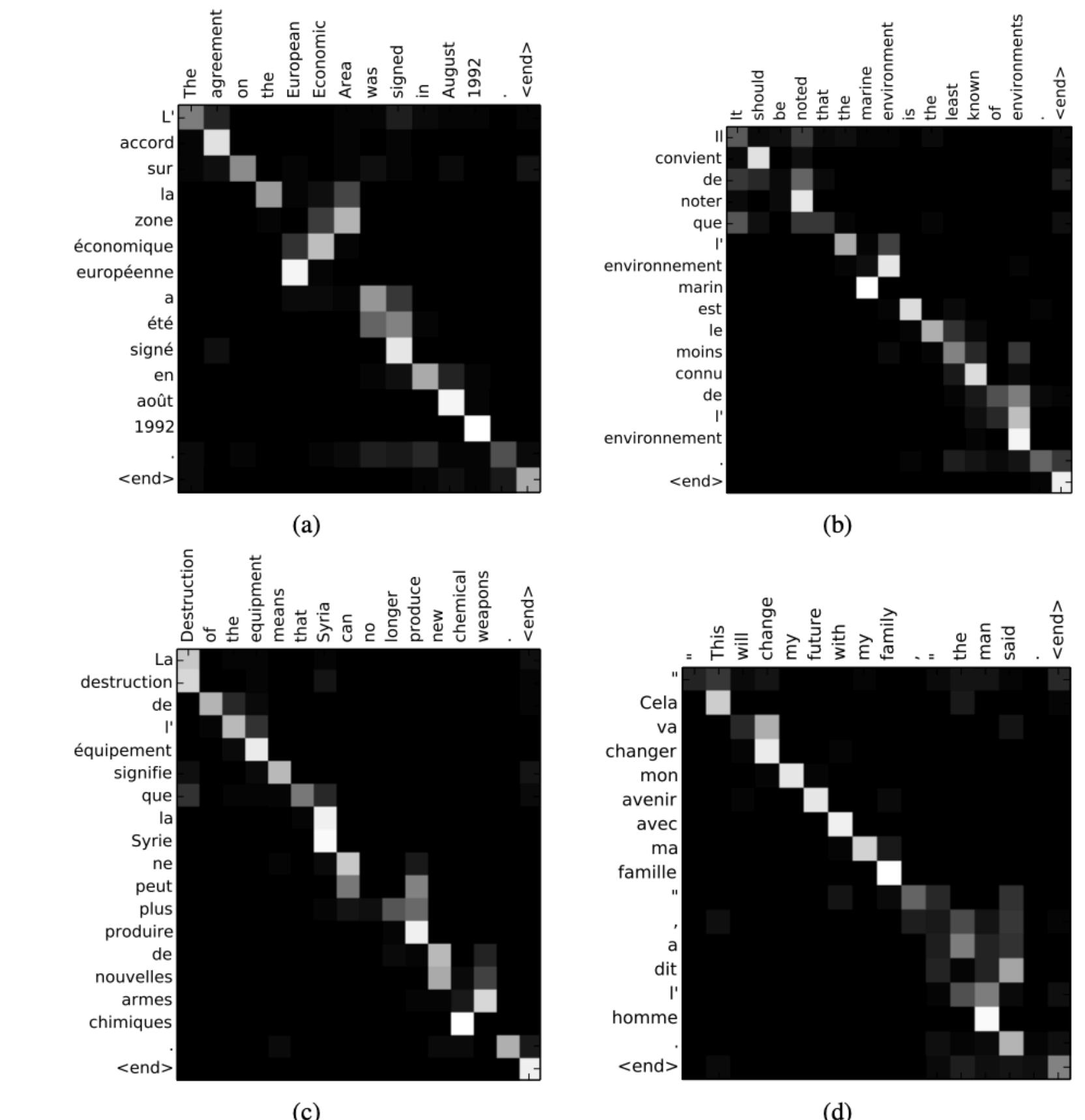


Figure 3: Four sample alignments found by RNNsearch-50. The x-axis and y-axis of each plot correspond to the words in the source sentence (English) and the generated translation (French), respectively. Each pixel shows the weight α_{ij} of the annotation of the j -th source word for the i -th target word (see Eq. (6)), in grayscale (0: black, 1: white). (a) an arbitrary sentence. (b-d) three randomly selected samples among the sentences without any unknown words and of length between 10 and 20 words from the test set.



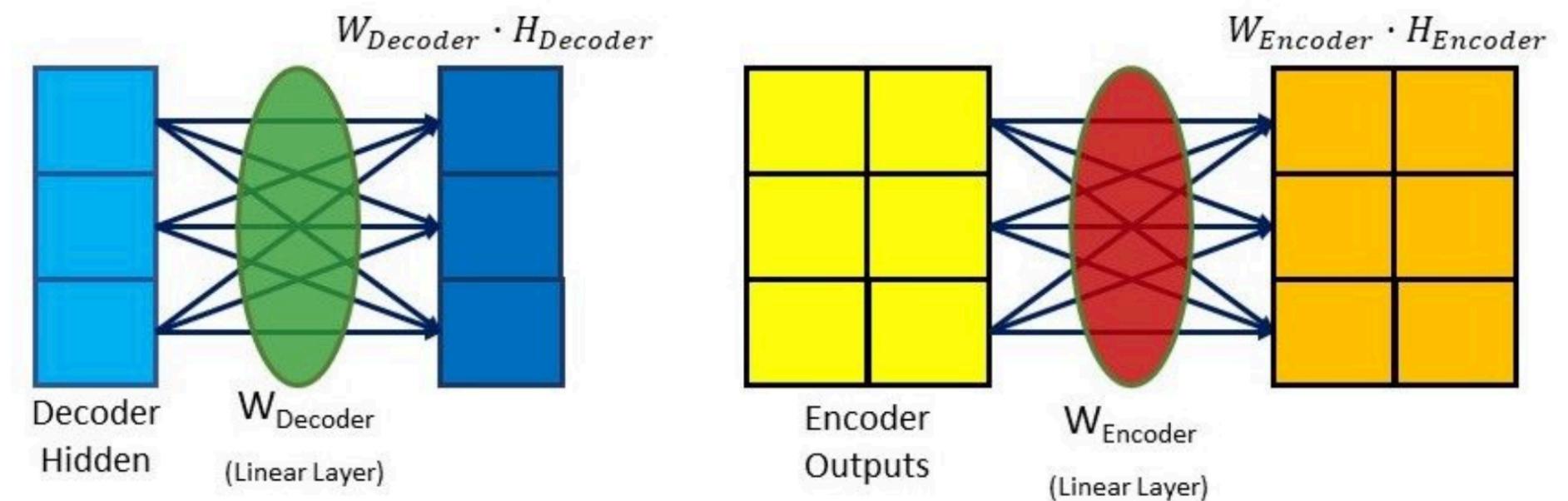
Alignment score

アライメントスコア

アライメントスコアの計算には、次の二種類の方法があります。

Bahdanau Attention

アライメントスコア

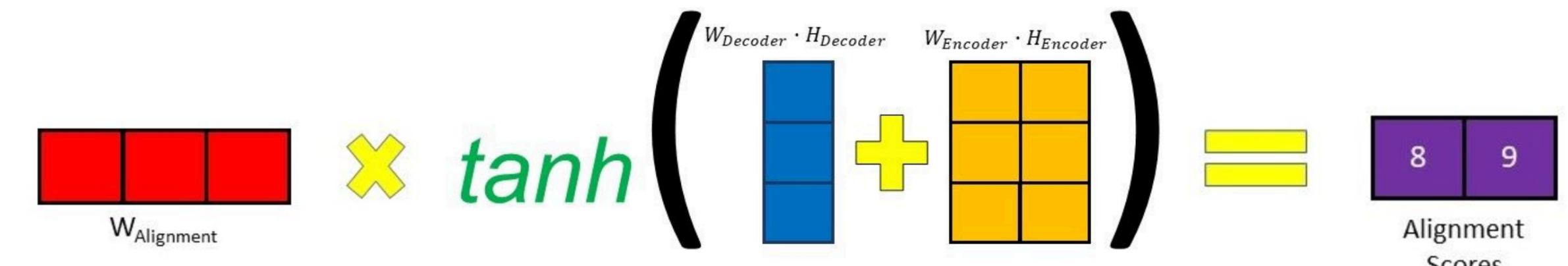


Bahdanau Attention

(Additive Attention : 加法Attention)

$$\text{score}(s_t, h_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[s_t; h_i])$$

こちらはDzmitry Bahdanau, et al (2014) で提案された方法です。デコーダの隠れ状態とエンコーダの各出力を加法 (add) したものをTanh関数に通し、アライメントの重みとの内積を行います。この重みパラメータは学習によって最適化されます。



Bahdanau Attention

アテンションウェイト
コンテキストベクトル

$$\alpha_{ts} = \frac{\exp(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_s))}{\sum_{s'=1}^S \exp(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_{s'}))}$$

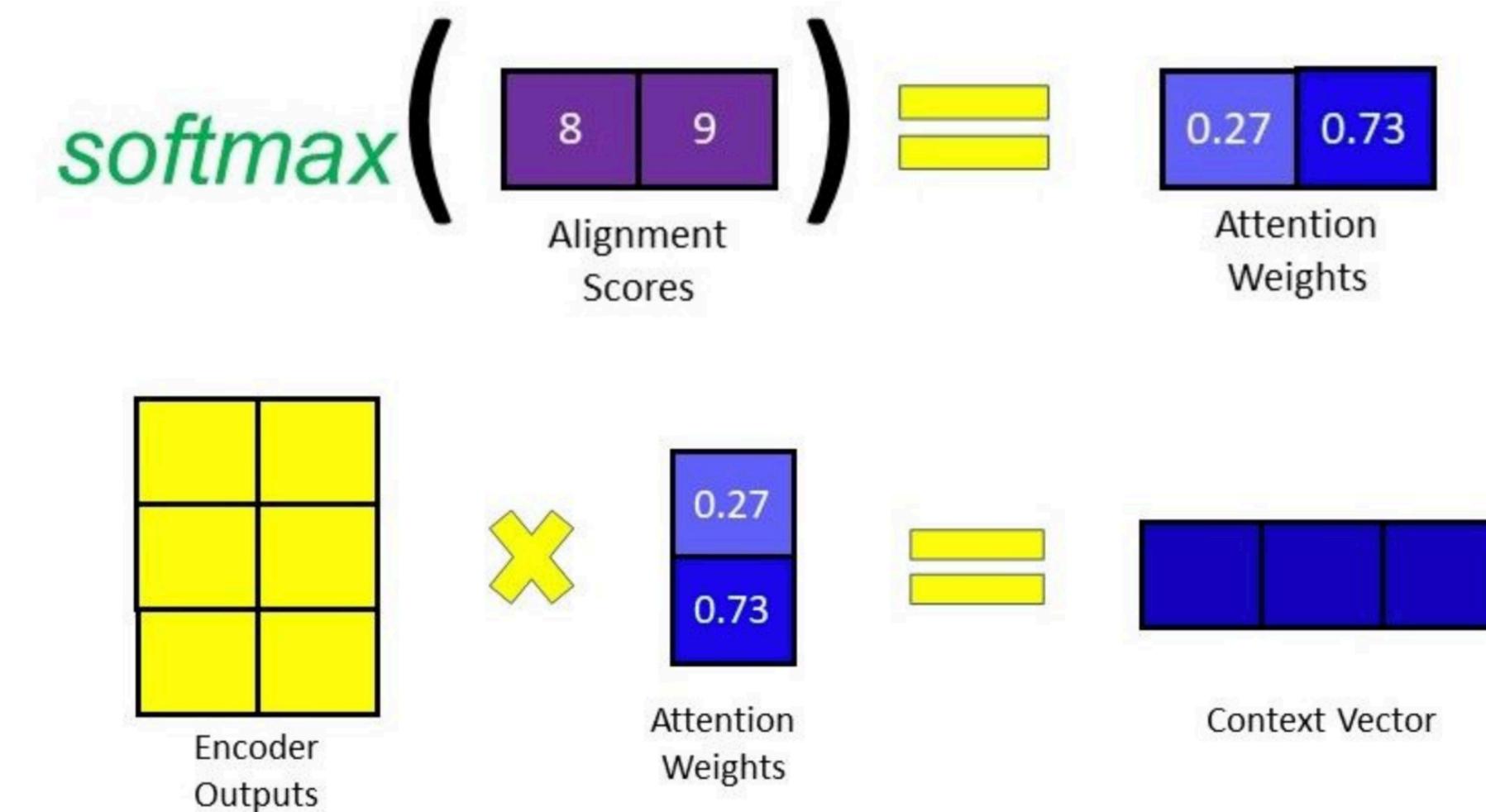
$$\mathbf{c}_t = \sum_s \alpha_{ts} \bar{\mathbf{h}}_s$$

[Attention weights]

[Context vector]

つまり、この重みをもつ隠れ層一つの全結合層を通して、①アライメントスコアを計算することになります。

さらに、アライメントスコアをsoftmaxに通し、エンコーダの隠れ状態との内積によって②コンテキストベクトルを作成します。



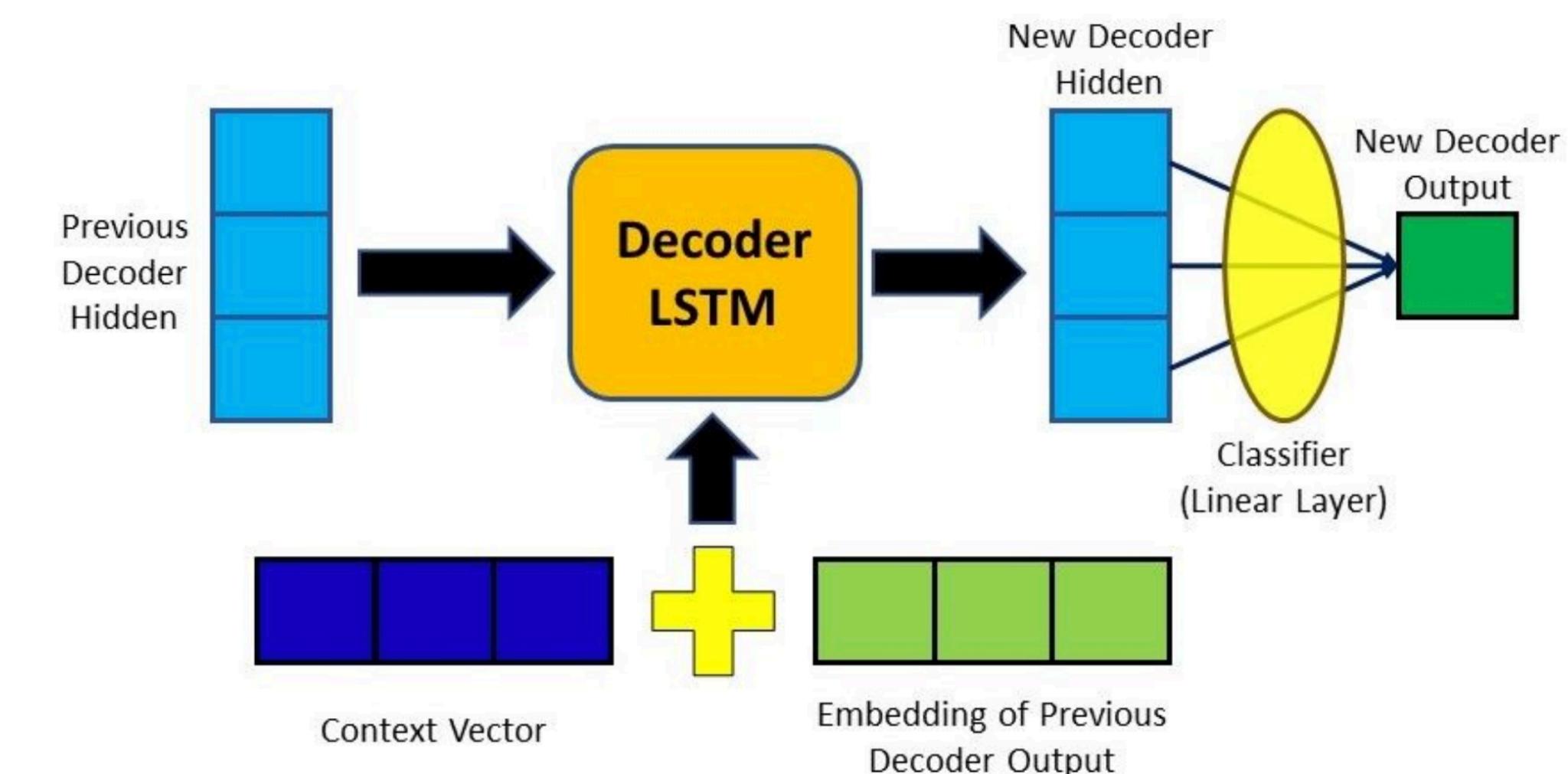
Bahdanau Attention

アテンションベクトル

$$\mathbf{a}_t = f(\mathbf{c}_t, \mathbf{h}_t) = \tanh(\mathbf{W}_c[\mathbf{c}_t; \mathbf{h}_t])$$

[Attention vector]

このコンテキストベクトルを、前のステップでのデコーダの出力に連結し (concatenate) 、現在のタイムステップのデコーダRNNに、ひとつ前の隠れ状態とともにに入力し、出力をデコードします。



Bahdanau Attention のクラス作成例:

https://www.tensorflow.org/tutorials/text/nmt_with_attention

Luong Attention

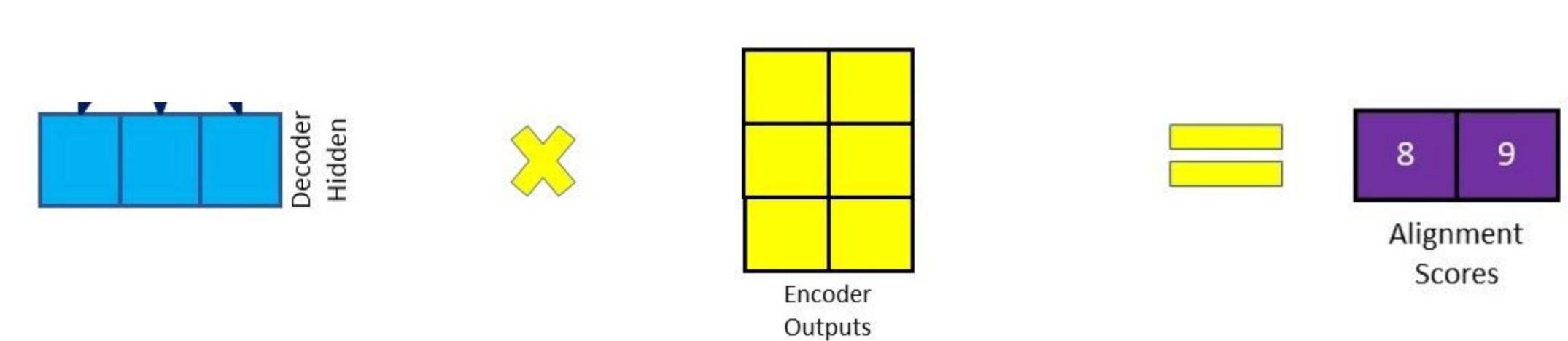
アライメントスコア

Luong Attention

(Dot-Product Attention : 内積Attention)

$$\text{score}(s_t, h_i) = s_t^\top h_i$$

デコーダの隠れ状態とエンコーダの各出力自体の内積によってアライメントスコアを計算します。ゆえに重みパラメータが必要なく、Bahdanau Attentionよりも計算時間を短縮できます。

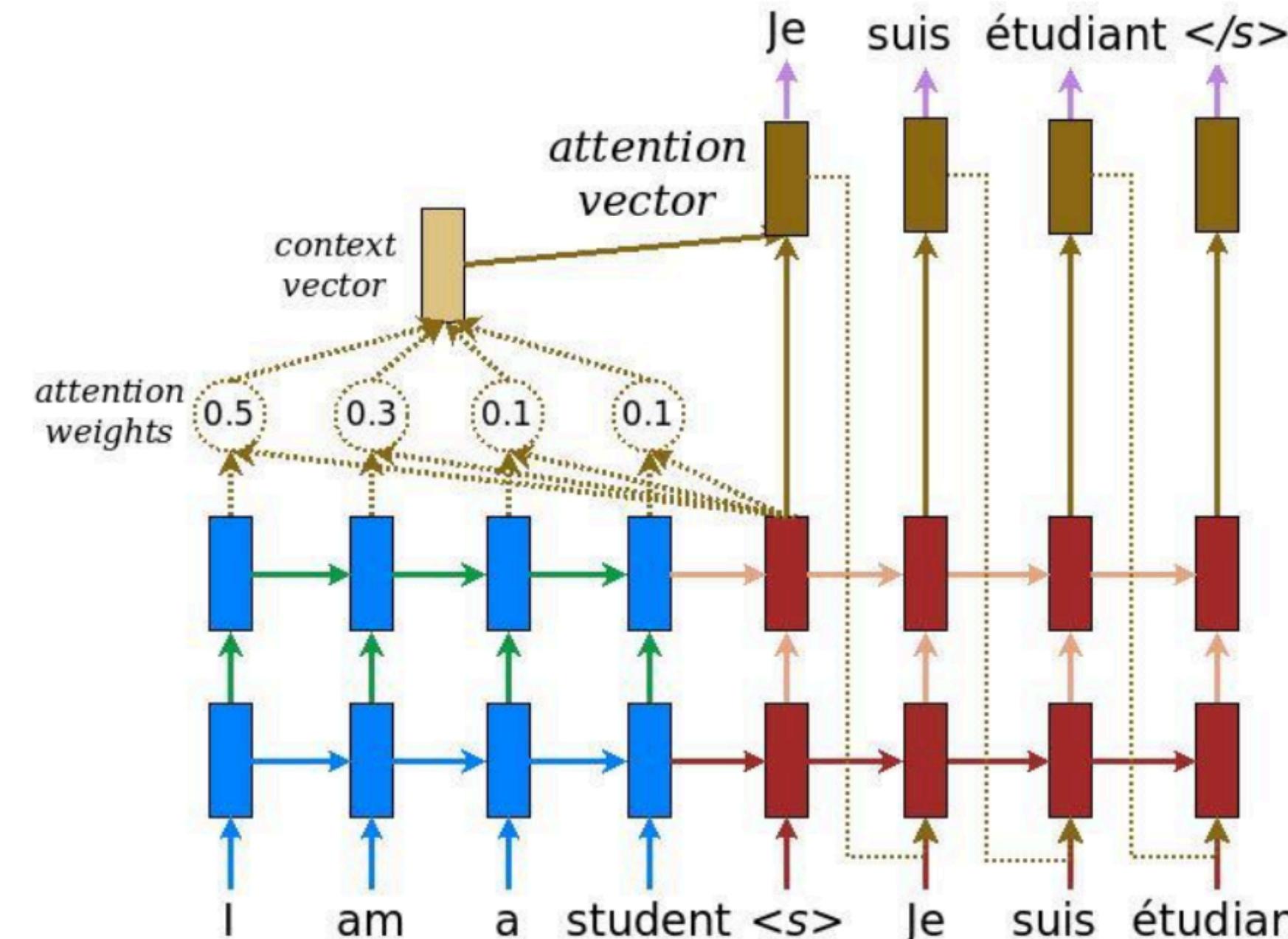


Luong Attention

Luong Attentionは、Bahdanauの手法を元に、Minh-Thang Luong, et al (2015) [3]において発表された三種類のアライメントスコアの計算方法のうちの一つに当たります。

アライメントスコアの計算方法と、コンテキストベクトルがデコーダに導入される位置が、Bahdanauのものと異なります。

Luongのコンテキストベクトルは前のタイムステップのデコーダの隠れ状態と連結し、そのタイムステップでの新しい出力を生成します。



TransformerでもこちらのLuong Attentionを踏襲しています。

[3]Minh-Thang Luong, Hieu Pham and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation.arXiv preprint arXiv:1508.04025v5, 2015