Attention Weights in Transformer NMT Fail Aligning Words Between Sequences but Largely Explain Model Predictions

Javier Ferrando and **Marta R. Costa-jussà**
TALP Research Center, Universitat Politècnica de Catalunya, Barcelona
{javier.ferrando.monsonis,marta.ruiz}@upc.edu

Abstract

This work proposes an extensive analysis of the Transformer architecture in the Neural Machine Translation (NMT) setting. Focusing on the encoder-decoder attention mechanism, we prove that attention weights systematically make alignment errors by relying mainly on uninformative tokens from the source sequence. However, we observe that NMT models assign attention to these tokens to regulate the contribution in the prediction of the two contexts, the source and the prefix of the target sequence. We provide evidence about the influence of wrong alignments on the model behavior, demonstrating that the encoder-decoder attention mechanism is well suited as an interpretability method for NMT. Finally, based on our analysis, we propose methods that largely reduce the word alignment error rate compared to standard induced alignments from attention weights.

1 Introduction

Recently, Transformer-based models (Vaswani et al., 2017) have allowed huge improvements in performance across multiple NLP tasks. The inclusion of this architecture has led the field of NLP to investigate the inner workings of this architecture in several tasks. One of its core components, the attention mechanism, which provides a distribution of scores over the input tokens, has been often presented as showing the relative importance of the inputs. Some works have criticized the use of attention weights as model explanations (Jain and Wallace, 2019; Serrano and Smith, 2019; Pruthi et al., 2020), demonstrating that attention weights distributions can be modified without affecting the final prediction. However, these studies have mainly analyzed encoder-only or decoder-only architectures like BERT (Devlin et al., 2019) or GPT-2 (Radford et al., 2019), which are based on self-attention mechanisms.

Nonetheless, NMT models use the encoder-decoder Transformer architecture, which adds the encoder-decoder attention mechanism, in charge of distributing the information flow from the encoder representations of the source input tokens into the decoder. (Voita et al., 2019) analyze the effect of pruning different attention heads in a Transformer NMT model and conclude that the encoder-decoder attention mechanism is the most critical one. (Raganato et al., 2020) show that encoder self-attention weights can be interchanged by predefined non-learnable patterns without hindering the translation performance. These results provide evidence about the relevance of the encoder-decoder attention mechanism on NMT, which we believe needs further investigation. In this work we analyze the encoder-decoder attention weights and shed light on their impact on the decoder representations and final predictions, showing how alignment errors can also give information about the model's decision-making process.

Research in NMT interpretability has mainly focused on understanding source words importance when predicting a target word. The word alignment task (Och and Ney, 2003) has served to compare explanation methods against human-annotated source-target word alignments. Encoder-decoder attention weights have been used to provide source-target word alignments (Zenkel et al., 2019; Garg et al., 2019), but its low performance has made researchers sceptical about its use as an interpretable method (Li et al., 2019). An important issue when relying on word alignment task is that it ignores the words that are predicted based on the target prefix, i.e what the model has previously translated. An extreme example of the impact of the target prefix on the prediction occurs during 'hallucinations' (Lee et al., 2019; Berard et al., 2019; Voita et al., 2020; Raunak et al., 2021). Although some studies have analyzed the relative contribution of the target prefix context in a model's prediction (Li et al.,

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 434–443
November 7–11, 2021.

2019; Voita et al., 2020), the way NMT models decide in which proportion to use both sequences remains unexplored.

In Section 3, we propose a simple method to measure the relative contribution of the source and the target prefix by perturbing input embeddings, we also extend the gradient-based method towards the target prefix token embeddings to obtain saliency scores from the prefix words. Both methods serve us in Section 5 to understand the attention weights generated in the encoder-decoder modules and their relationship with the model predictions. Lastly, in Section 6, we propose two methods to improve the alignment error extracted from the model in accordance with our results analysis.

2 Background

In this section, we briefly introduce the existing methodologies that we use in our work: the Transformer and the methods used to induce alignment.

2.1 Transformers in NMT

Given a source sequence $\mathbf{x} = \{x_1, \cdots, x_{|\mathbf{x}|}\}$[1] and a target sequence $\mathbf{y} = \{y_1, \cdots, y_{|\mathbf{y}|}\}$ an NMT system models the probability:

$$P(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{|\mathbf{y}|} P(y_t|\mathbf{y_{<t}}, \mathbf{x})$$

where $\mathbf{y_{<t}} = \{y_0, \cdots, y_{|t-1|}\}$ represent the prefix of y_t, and $x_{|\mathbf{x}|} = y_0 = y_{|\mathbf{y}|} = \langle / \mathrm{s} \rangle$, which represents a special token used to denote the beginning and end of sentence. The Transformer architecture is composed by a stack of encoder layers and decoder layers. The encoder generates a contextualized sequence of representations $\mathbf{e} = \{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_{|\mathbf{x}|}\}$ of the source sentence while the decoder, at each time step t, uses both the encoder output and the token representation $\boldsymbol{s}_t^{l-1}$ of the previous layer l to compute the final probability distribution over the target vocabulary.

In terms of the model's input and output, we consider $\boldsymbol{x}_j$ and $\boldsymbol{y}_i$ representing the embeddings of each token from the source and target prefix respectively. So, we can write the conditional probability modeled by the network at each time step as:

$$P(y_t|\{\boldsymbol{y}_0, \cdots, \boldsymbol{y}_{t-1}\}, \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{|\mathbf{x}|}\})$$

[1]Along this work we use x to represent elements (scalars/words/tokens), $\boldsymbol{x}$ vectors, $\mathbf{x}$ sequences and $\mathbf{X}$ matrices.

The encoder and decoder representations are merged in the multi-head encoder-decoder attention mechanism (Figure 1). For each head, the encoder embeddings are projected to keys and values. Formally, $\mathbf{V}^h \in \mathbb{R}^{|\mathbf{x}| \times d_v}$ is the value matrix and $\mathbf{K}^h \in \mathbb{R}^{|\mathbf{x}| \times d_k}$ is the key matrix, where d_v and d_k refers to the dimension of the values and keys vectors. The decoder representation of the output token $\boldsymbol{s}_t^{l-1}$ is projected to a query vector of dimension d_q, $\boldsymbol{q}_t^h \in \mathbb{R}^{d_q}$. The output of each attention head is obtained by:

$$\boldsymbol{z}_t^h = \sum_{j=1}^{|\mathbf{x}|} \boldsymbol{\alpha}_{t,j}^h \boldsymbol{v}_j^h \tag{1}$$

Where:

$$\boldsymbol{\alpha}_t^h = \text{softmax}\left(\frac{\boldsymbol{q}_t^h \mathbf{K}^\top}{\sqrt{d_k}}\right)$$

$\boldsymbol{\alpha}_t^h$ refers to the vector of attention scores at decoding step t, which is often presented as a matrix (attention matrix) made of a stack of $\boldsymbol{\alpha}_t^h$, for every time step. This process is repeated simultaneously in multiple heads. Each head computes a $\boldsymbol{z}_t^h$ representation, and are concatenated before projecting by $\mathbf{W_h^O}$ to obtain $\boldsymbol{attn}_t$.

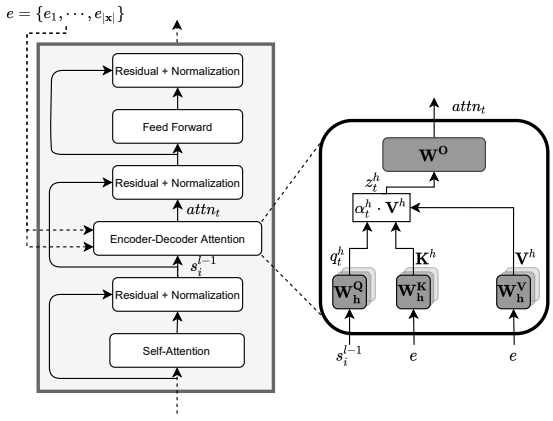

Figure 1: Decoder Layer with the Encoder-Decoder Attention module expanded.

2.2 Attention Weights to Induce Word Alignment

Attention weights $\alpha_{t,j}^h$ from the encoder-decoder attention modules represent the similarity between $\boldsymbol{h_j}$ and $\boldsymbol{s}_t^{l-1}$ and have been commonly presented as a baseline to extract word alignments from words x_j and y_t. Attention vectors $\boldsymbol{\alpha}_t^h$ represent a probability distributions over all source tokens $\mathbf{x}$. A

classical approach to obtain final alignments has been to compute the average over all heads (Garg et al., 2019) in each layer and selecting the source word that yields the maximum score:

$$\mathbf{A}_{t,j} = \begin{cases} 1 & j = \arg\max_{j'} \frac{1}{H} \sum_{h=1}^{H} \alpha_{t,j'}^{h} \\ 0 & \text{else} \end{cases}$$

(Zenkel et al., 2019; Li et al., 2019; Garg et al., 2019) showed alignments induced from attention weights are noisy, although they realize that some layers seem to generate better (x_j, y_t) alignments, especially the last layers of the Transformer.

An issue regarding the use of this method to interpret the model predictions is that the ground truth target word y_t may differ from the actual model prediction y'_t. In these cases, (x_j,y'_t) alignments can not be compared with (x_j,y_t) gold alignments, showing limitations about its use as an interpretability method.

Aligments from the decoder input. A technique that solves the aforementioned issue consists of inducing alignments by comparing $\mathbf{x}$ with the input of the decoder y_i (Kobayashi et al., 2020; Chen et al., 2020) (in force decoding setting $y_i = y_{t-1}$). So, since the ground truth target sequence is used as input in the decoder, alignments $\mathbf{A}_{i,j}$ in this setting represent the same information as gold alignments. Attention modules from the initial layers tend to extract better alignments from the input of the decoder, while alignments from the decoder output are better extracted from the final layers. Although results show that decoder input provides lower alignment error rates, it shows how similar to $\boldsymbol{e}_j$ the model is able to generate representations of the decoder input, losing explanation power about the influence of source tokens into the model output. Therefore, we use $\mathbf{A}_{t,j}$ in our analysis in Section 5. An extension of the use of attention weights to induce alignments is presented in (Kobayashi et al., 2020), where it is also considered the norm of the vectors projected by the linear layers inside the attention modules.

2.3 Other Methods

Model-agnostic methods. Several methods for inducing alignments have been proposed that work regardless of the chosen architecture. Gradient-based methods such as gradient $\times$ input (Ding et al., 2019) or Integrated Gradients (He et al., 2019) have been used to obtain saliency values from the source words as a measure of source word importance. Erasure methods have also been applied to NMT (Li et al., 2019), which consist of techniques to measure the relevance of each input token by evaluating the changes in the output probability of the model after removing it from the input of the network (Zintgraf et al., 2017) or eliminating the connection via dropout (Srivastava et al., 2014).

Methods to improve alignments. Other works propose methods to improve word alignment extracted from the Transformer. (Li et al., 2019) use an explicit alignment model (Liu et al., 2005; Taskar et al., 2005) consisting of optimizing a parameter matrix to reduce the alignment distance with respect to a reference. (Zenkel et al., 2019) adds an alignment module attending encoder representations. (Garg et al., 2019) propose to supervise an attention head with GIZA++ (Brown et al., 1993) alignments. Although they improve alignment performance, these methods introduce external trainable parameters or alignments references, which makes these techniques lose interest regarding interpretability of the model.

3 Proposed Methods for Analysis

In this section, we introduce two simple methods for measuring the contributions of each source sequence to a model prediction and extend the gradient-based analysis to understand dependency relationships between target prefix words.

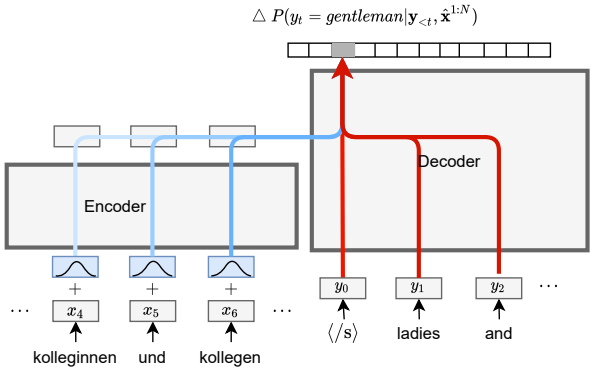

Figure 2: Contribution from the source input sequence to the final prediction by perturbing source token embeddings.

3.1 Contributions by Input Perturbation

We propose separately perturbing source and prefix embeddings (Smilkov et al., 2017) to get the marginal contributions of each sequence to the final prediction. For each embedding we compute

N random samples around their neighborhood:

$$\hat{\boldsymbol{x}}_j = \boldsymbol{x}_j + \mathcal{N}(0, \sigma^2_{\boldsymbol{x}_j})$$

Since input embeddings differ in their length, we adapt the noise level to each token embedding as a proportion (λ) of its euclidean norm[2]:

$$\sigma_{\boldsymbol{x}_j} = \|\boldsymbol{x}_j\| \cdot \lambda$$

By adding noise to each embedding in the sequence we get the perturbed sequence of embeddings $\hat{\mathbf{x}}$. So, for each prediction y_t we can compute the source contribution $C_S(y_t)$ by measuring how large is the variation of the output probability when feeding the network with N noisy sequence samples. To get the marginal effect of one sequence, we keep the other with the original embeddings.

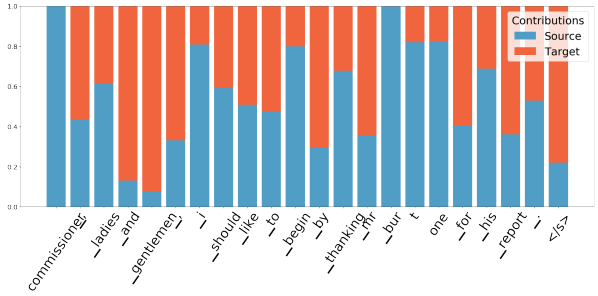

Figure 3: Source $C_S(y_t)$ and target prefix $C_T(y_t)$ contributions for reference model output.

$$C_S(y_t) = \triangle P(y_t|\mathbf{y}, \hat{\mathbf{x}}^{1:N})$$
$$= \frac{1}{N}\sum_{n=1}^{N}(P(y_t|\mathbf{y}_{<t}, \hat{\mathbf{x}}^n) - \bar{P}(y_t|\mathbf{y}_{<t}, \hat{\mathbf{x}}^{1:N}))^2$$

where $\bar{P}$ refers to the mean of the observed output probabilities and $\hat{\mathbf{x}}^n$ to n-th source sequence with added noise. Similarly, we get the target prefix contribution $C_T(y_t)$ perturbing prefix embeddings:

$$\hat{\boldsymbol{y}}_i = \boldsymbol{y}_i + \mathcal{N}(0, \sigma^2_{\boldsymbol{y}_i})$$

and then, computing the variance of the output probability across N sequences of noisy prefix embeddings, keeping untouched the original source token embeddings:

$$C_T(y_t) = \triangle P(y_t|\hat{\mathbf{y}}^{1:N}_{<t}, \mathbf{x})$$
$$= \frac{1}{N}\sum_{n=1}^{N}(P(y_t|\hat{\mathbf{y}}^{n}_{<t}, \mathbf{x}) - \bar{P}(y_t|\hat{\mathbf{y}}^{1:N}_{<t}, \mathbf{x}))^2$$

[2]In this work we use $\lambda = 1\%$

3.2 Saliency of Target Sequences Words

Any model $f(\boldsymbol{x})$ can be linearly approximated locally by its first-order Taylor expansion at a point $\hat{\boldsymbol{x}}$:

$$f(\hat{\boldsymbol{x}}) \approx f(\boldsymbol{x}) + \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) \cdot (\hat{\boldsymbol{x}} - \boldsymbol{x})$$

Rearranging terms we get:

$$f(\boldsymbol{x}) \approx f(\hat{\boldsymbol{x}}) + \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) \cdot (\boldsymbol{x} - \hat{\boldsymbol{x}})$$

Making $\hat{\boldsymbol{x}}$ a zero vector, we arrive to:

$$f(\boldsymbol{x}) \approx \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) \cdot \boldsymbol{x}$$

With this approximation, $\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$ can be interpreted as coefficients that measure the impact of $\boldsymbol{x}$ in the output. In NLP (Li et al., 2016) propose the use of word embeddings as input features from which to calculate saliency scores. In the NMT setting, current methods (Ding et al., 2019) extract saliency scores of the input source tokens by computing the gradient with respect to source embeddings $\boldsymbol{x}_i$.

Nevertheless, the Transformer model deals with two different sequences of inputs ($\mathbf{x}$ and $\mathbf{y}_{<\mathbf{t}}$), $f(\boldsymbol{x}) = P(y_t|\mathbf{y}_{<\mathbf{t}}, \mathbf{x})$. So, analyzing only the saliency of the source sequence embeddings might lead to an incomplete analysis. To have a full understanding of the influences of each input word on the model prediction we propose to extend the SmoothGrad method (Smilkov et al., 2017) to also consider the gradients w.r.t the target prefix embeddings. We compute the target prefix saliencies by averaging the gradients over N noisy examples, as detailed in Section 3.1:

$$\psi(y_i, y_t) = \frac{1}{N}\sum_{n=1}^{N}\|\nabla_{\boldsymbol{y}_i} P(y_t|\hat{\mathbf{y}}^n_{<t}, \mathbf{x})\|$$

4 Experimental Setup

As follows, we detail the model and datasets used in our experiments. We decide to choose this experimental framework to compare and further explain previous works (Ding et al., 2019; Zenkel et al., 2019; Kobayashi et al., 2020). We follow the same procedure as these past works, we train the Transformer model for the German-English translation task. Specifically, we use Europarl v7 corpus[3] which consists on 1.9M sentence pairs. We use the

[3]http://www.statmt.org/europarl/v7

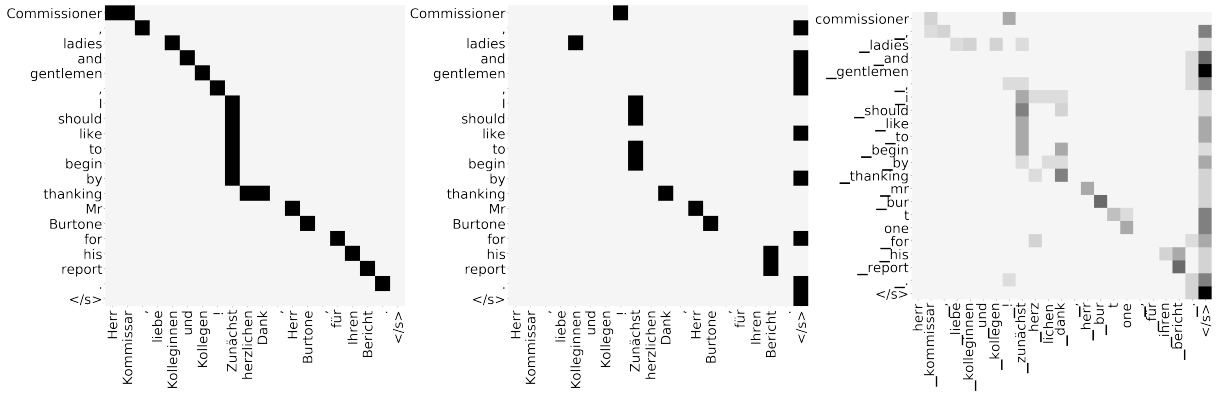

Figure 4: From left to right: gold alignments, hard alignments $\mathbf{A}_{t,j}$ and soft alignments given by the average attention weights over all heads in Layer 5.

gold alignment dataset[4] (Vilar et al., 2006) which contains 508 sentence pairs. The Transformer used in this work [5] is implemented in fairseq (Ott et al., 2019) and contains 6 layers with 4 attention heads each. We apply Byte Pair Encoding (BPE) (Sennrich et al., 2016) with 10k merging operations. As (Ding et al., 2019) and (Kobayashi et al., 2020) we use the last 1000 samples of the training data as the development data.

5 Analysis

In this section, and for the sake of clarity, we convey our analysis on a single example while quantifying how our findings generalize to the entire test set. We use the following source example:

(1) *Herr Kommissar, liebe kolleginnen und kollegen! Zunächst herzlichen Dank, Herr Burtone, für Ihren Bericht.*

for which the model prediction is:

(2) *Mr, ladies and gentlemen, first would like to start by thanking Mr Burtone for his report.*

and with its reference[6]:

(3) *Commissioner, ladies and gentlemen, I should like to begin by thanking Mr Burtone for his report.*

[4] https://www-i6.informatik.rwth-aachen.de/goldAlignment/

[5] transformer_iwslt_de_en

[6] We refer to the words that are predicted with the highest probability as the model prediction y'_t, and the reference (ground truth) words as the ground truth or reference model output y_t.

5.1 Categorization of Word Alignment Errors

Figure 4 (Middle) shows (x_j, y_t) alignments $\mathbf{A}_{t,j}$ extracted from the best layer (§2.2) for the example. Figure 4 (Right) depicts the average attention weight matrix across all heads in the best layer (soft alignments), from which some information can be recovered. When comparing hard with gold alignments (Figure 4 (Left)), some errors are clearly observed, with a large number of target tokens aligning to finalizing tokens. Hereinafter, finalizing tokens correspond to the special token used to indicate end of sentence ($\langle$/s$\rangle$) and the final punctuation mark (_.), while the rest of tokens will be referred to as *standard tokens*.

We categorize alignment errors occurring in weight attention matrices as:

1. Functional/content words aligning to finalizing tokens.
2. Words with non-direct translation aligning to finalizing tokens.
3. Last tokens of a split word (divided into multiple subwords) aligning to finalizing tokens.
4. Functional words aligning to the next content word token.
5. Words aligning to other standard tokens.

Our analysis focuses on finding explanations to the errors inside categories 1-3, which account for 60.6% and 38.7% of the total errors in the best layer in the (x_j, y_t) and (x_j, y_i) alignment settings respectively.

5.2 Encoder-Decoder Attention Module decides Source-Target Contributions

From source-target contributions of the reference model output (Figure 3), we observe that the target prefix largely contributes when predicting *gentlemen*, and it also receives large saliency scores (Figure 5) from *ladies*. This matches the human intuition about how these words are naturally generated from the context.

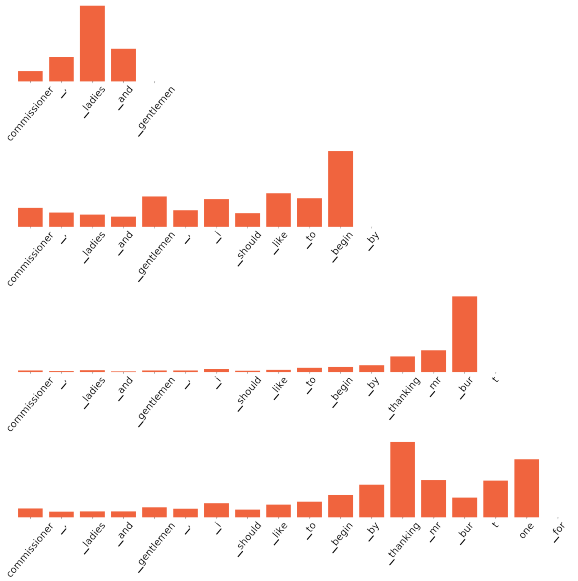

Figure 5: Saliency scores $\psi(\mathbf{y}_{<t}, y_t)$ for the reference model output (from top to bottom): `_gentlemen`, `_by`, `_t` and `_for`.

Similarly, a non-common word such as *Burtone*, which gets tokenized into `_bur`, `t` and `one` gets source-target contributions that also match human intuition. The first token `_bur` is predicted by relying almost only on the source sequence. However, following tokens, although they heavily rely on the source, get information about the previous tokens. In this case, `_bur` gives a high saliency value when predicting `t`. We can also observe that the word *by* is mainly predicted using target prefix, and gets the highest saliency score from *begin*. Another observation is that *thanking* highly influences the prediction of *for*. These examples have in common both large dependency on the target prefix and large attention values towards finalizing tokens.

Model behaviour. From the decoder layer depicted in Figure 1 we can observe that the output of the encoder-decoder attention module is added to the target prefix representation by means of the residual connection $\boldsymbol{attn}_t + \boldsymbol{s}_t^{l-1}$. Therefore, the amount of information arriving from the input sequence is determined by the weighted sum of the values. If we analyze the norms of the values vectors (Figure 6) we can see that the source finalizing tokens, especially $\langle$`/s`$\rangle$, get almost zero norms. This can be interpreted as when assigning high attention weights to these tokens, the Residual + Normalization layer gets almost no information from the source. From the results obtained over 5 random seeds (Figure 7) we can state that the network picks a common token, i.e. $\langle$`/s`$\rangle$ or `_.` and projects it to a zero vector through $\mathbf{W_h^V}$. These results support the (Clark et al., 2019) hypothesis about the selection of a token as a "no-op" in the attention mechanism (`[SEP]` token in BERT model).

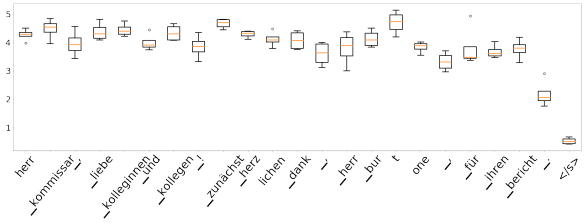

Figure 6: $\|v_j^h\|$ computed in every attention head. $\langle$`/s`$\rangle$ has almost zero norm for every head.

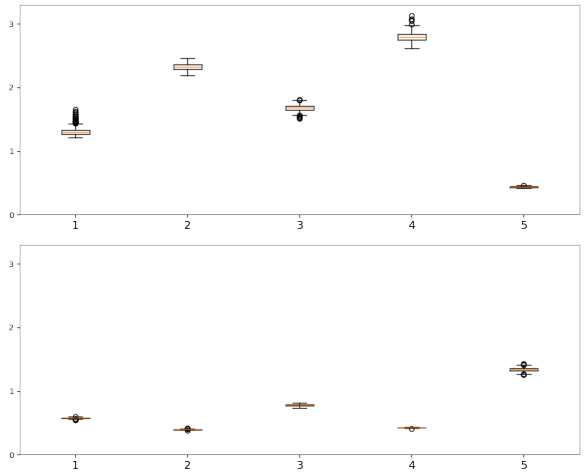

Figure 7: $\|v_j^h\|$ for `_.` (Top) and $\langle$`/s`$\rangle$ (Bottom) for the best alignment head over 5 random seeds.

In this way, by putting attention to it, decides how much amount of information flows from the source and target sequences. Note that over the five trained models, the selection of the token that ends up squished varies, for model number 5, the network selects the final punctuation mark as the token used to cancel source contribution. $\boldsymbol{attn}_t$ vector norms (Figure 8) correlate with our source-

target contribution method results depicted in Figure 3. Representations $\boldsymbol{attn}_t$ for tokens such as `and`, `gentlemen` and `_,` have low norms due to the effect of large attention weights towards finalizing tokens.

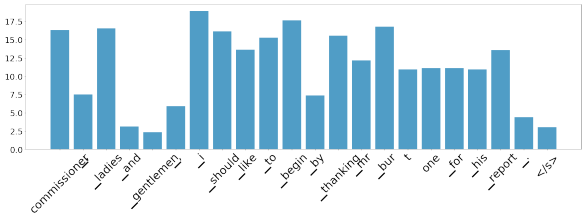

Figure 8: Output representation of the encoder-decoder attention module norms $\|\boldsymbol{attn}_t\|$.

Interestingly, the finalizing tokens representations from the last encoder layer show clear differences with respect to the other tokens' representations. Measuring the cosine similarity between every encoder output representation (Figures 9 and 10) we observe how the finalizing tokens similarity with every other encoder representation is consistently negative.

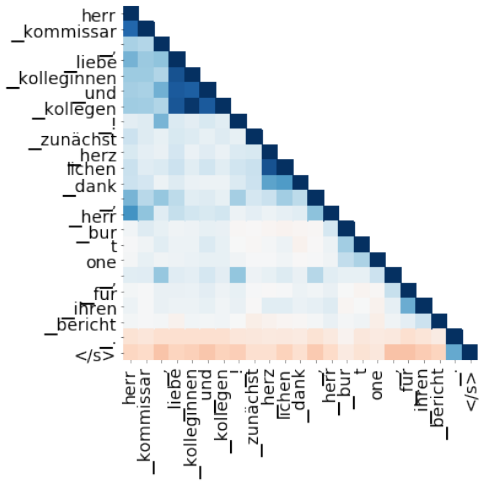

Figure 9: Cosine similarity between encoder representations. Positive similarity (blue), negative (red).

We conjecture that these tokens encode minimum information about the source sentence, and the decoder finds them useful in the encoder-decoder attention module to skip source attention. We leave as future work a deeper investigation of this phenomenon.

5.3 Word Alignment Errors associated to Part-of-Speech

If we analyze the percentage of tokens across the whole dataset that are aligned towards finalizing

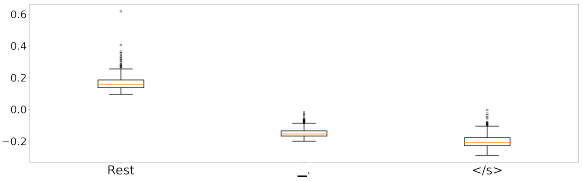

Figure 10: Cosine similarity between encoder representations by type of token across the test set.

tokens, i.e receiving attention scores greater than 0.5, we observe (Table 1) that words with a high degree of dependency on the context such as adpositions (ADP), particles (PART) and conjunctions (SCONJ, CCONJ) are likely to get aligned to finalizing tokens. On the other hand, numerical values (NUM), determiners (DET) and verbs (VERB, AUX), which are more independent of the context tend to align to source tokens. We see that functional words are more prone to get aligned to finalizing tokens.

POS-tag	%
ADP	49.1
PART	33.9
SCONJ	30.7
NOUN	24
CCONJ	23.8
ADV	17.3
PROPN	16
PRON	14
ADJ	13.1
VERB	12.4
DET	9.4
NUM	6.8
AUX	4.5

Table 1: Words (in %) aligning to finalizing tokens.

These results agree with our previous observations. Words with a high contribution from the target prefix get attention weights assigned to source finalizing tokens. These results demonstrate a correlation between the attention towards finalizing tokens and the lack of contribution from the source to the model prediction.

6 Methods to Improve Alignment

As shown in the previous analysis, alignments from attention weight matrices reveal errors mainly due to the existence of the skip source attention operation. In this section we propose two methods to get more clear alignments.

6.1 Heads Importance

Each layer attention weight matrix is computed by averaging over every head (§2.2). However,

we know specific heads learn better alignments (Kobayashi et al., 2020). Based on the Hidden Token Attribution method (Brunner et al., 2020) we measure the contribution of each head to the output of the model and detect that specialized heads tend to obtain higher contributions. We propose to optimize the extraction of per layer attention weights substituting the naive average approach by a weighted average based on each head contribution. For each head h we compute the summation of the gradients w.r.t the input vectors $\boldsymbol{v}_j^h$:

$$c_h(y_t) = \sum_{j=1}^{|\mathbf{x}|} \left\| \nabla_{\boldsymbol{v}_j^h} P(y_t|\mathbf{y}_{<t}, \mathbf{x}) \right\|$$

Then, to extract its relative contribution, we normalize between the scores of every head:

$$C_h(y_t) = \frac{c_h(y_t)}{\sum_{h=1}^{H} c_h(y_t)}$$

Finally, we extract hard alignment as a weighted average of the head's relative contribution:

$$\mathbf{A}_{t,j} = \begin{cases} 1 & j = \arg\max_{j'} \sum_{h=1}^{H} C_h(y_t) \alpha_{t,j'}^h \\ 0 & \text{else} \end{cases}$$

6.2 Masking Finalizing Tokens

Attention shifting towards source finalizing tokens make the models underperform in the Word Alignment Task. From our previous analysis we also demonstrate they are used to manage the amount of information from the prefix that flow to upper layers. We propose to mask the attention weights to the finalizing tokens with zeros to measure the degree of success of secondary attention weights induced alignments.

6.3 Modified Alignments Results

Results in Table 2 reflect the reduction in alignment error rate (AER) by applying the proposed methods. Regarding the heads importance method, it improves AER percentage in 2.7 points in the decoder input ($\mathbf{A}_{i,j}$) alignment setting, although maintaining same accuracy in the decoder output ($\mathbf{A}_{t,j}$) alignments. The difference in improvements in $\mathbf{A}_{i,j}$ are explained by the fact that initial layers have attention heads more specialized, while in the last layers they perform more uniformly. Masking methods reduces 6.3 and 8.4 AER points in $\mathbf{A}_{i,j}$ and $\mathbf{A}_{t,j}$ respectively, which indicates that, despite deciding that the prefix contributes the most, the model still pays attention to relevant source tokens.

Method	AER	± SD	AER	± SD
	$\mathbf{A}_{i,j}$		$\mathbf{A}_{t,j}$	
Attention weights				
(Kobayashi et al., 2020)	29.8	3.7	47.7	1.7
Ours (HI)	27.1	2.0	47.6	1.6
Ours (Mask)	23.5	1.1	39.3	1.5
Ours (HI + Mask)	22.1	1.2	38.5	1.7
(Chen et al., 2020)	20.9	-	-	-
Vector-Norms				
(Kobayashi et al., 2020)	25.0	1.5	41.4	1.4
Word Aligner				
Fast-Align	28.4	-	28.4	-
GIZA++	21.0	-	21.0	-

Table 2: AER results comparison. Our methods are applied on (Kobayashi et al., 2020) implementation, which we use as the reference. HI refers to the Heads Importance method (§6.1). GIZA++ and Fast-Align results from (Zenkel et al., 2019).

7 Conclusion

In this paper, we have studied the use of attention weights as an explanatory method for the Transformer in NMT. We have proposed analysis methods that measure the relative contribution of the source and the target prefix sequences. Then, we have demonstrated that the alignment bias towards finalizing tokens, which is the most common alignment error, is used by the model to avoid source information flowing through the decoder. In these cases, the predicted output relies on prefix dependencies, which are identifiable by extending the gradient-based analysis to extract saliency scores. Furthermore, we have proposed two methods to improve the extraction of alignments from attention weights. As future work, we plan to extend our study to more languages pairs, as well as to the multilingual NMT setting.

Acknowledgements

We would like to thank Ioannis Tsiamas for the help in the busy days, as well as Carlos Escolano and Christine Basta for the useful comments. This work is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 947657).

References

Alexandre Berard, Ioan Calapodescu, and Claude Roux. 2019. Naver labs Europe's systems for the WMT19 machine translation robustness task. In *Proceedings of the Fourth Conference on Machine*

Translation (Volume 2: Shared Task Papers, Day 1), pages 526–532, Florence, Italy. Association for Computational Linguistics.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On identifiability in transformers. In *International Conference on Learning Representations*.

Yun Chen, Yang Liu, Guanhua Chen, Xin Jiang, and Qun Liu. 2020. Accurate word alignment induction from neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 566–576, Online. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Shuoyang Ding, Hainan Xu, and Philipp Koehn. 2019. Saliency-driven word alignment interpretation for neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 1–12, Florence, Italy. Association for Computational Linguistics.

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly Learning to Align and Translate with Transformer Models. *arXiv:1909.02074 [cs]*. ArXiv: 1909.02074.

Shilin He, Zhaopeng Tu, Xing Wang, Longyue Wang, Michael Lyu, and Shuming Shi. 2019. Towards understanding neural machine translation with word importance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 953–962, Hong Kong, China. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. 2019. Hallucinations in neural machine translation.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, California. Association for Computational Linguistics.

Xintong Li, Guanlin Li, Lemao Liu, Max Meng, and Shuming Shi. 2019. On the word alignment from neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1293–1303, Florence, Italy. Association for Computational Linguistics.

Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 459–466, Ann Arbor, Michigan. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2020. Learning to deceive with attention-based explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4782–4793, Online. Association for Computational Linguistics.

A. Radford, Jeffrey Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. In *OpenAI Blog*.

Alessandro Raganato, Yves Scherrer, and Jörg Tiedemann. 2020. Fixed encoder self-attention patterns in transformer-based machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 556–568, Online. Association for Computational Linguistics.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda B. Viégas, and Martin Wattenberg. 2017. Smoothgrad: removing noise by adding noise. *CoRR*, abs/1706.03825.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Ben Taskar, Simon Lacoste-Julien, and Dan Klein. 2005. A discriminative matching approach to word alignment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 73–80, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

David Vilar, Maja Popovic, and H. Ney. 2006. Aer: do we need to "improve" our alignments? In *IWSLT*.

Elena Voita, Rico Sennrich, and Ivan Titov. 2020. Analyzing the Source and Target Contributions to Predictions in Neural Machine Translation. *arXiv:2010.10907 [cs]*. ArXiv: 2010.10907.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2019. Adding interpretable attention to neural translation models improves word alignment. *CoRR*, abs/1901.11359.

Luisa M. Zintgraf, Taco S. Cohen, Tameem Adel, and Max Welling. 2017. Visualizing deep neural network decisions: Prediction difference analysis. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.