1. Training Details

Data Pre-processing We mainly follows the pre-processing pipeline \(^1\) which is also adopted by (Ott et al., 2018), (Edunov et al., 2018) and (Zhu et al., 2020), except for the sub-word tool. To verify the consistency between the word piece model (Schuster & Nakajima, 2012) and the BPE model (Sennrich et al., 2016), we conduct a comparison experiment to train two standard Transformer models using the same data set processed by the word piece model and the BPE model respectively. The BLEU difference between them is about ±0.2, which suggests there is no significant difference between them.

Batching Data Transformer groups training examples of similar lengths together with a varying batch size for training efficiency (Vaswani et al., 2017). In our approach, when interpolating two source sentences, \(x^p\) and \(y^\circ\), it is better if the lengths of \(x^p\) and \(y^\circ\) are similar, which can reduce the chance of wasting positions over padding tokens. To this end, in the first round, we search for monolingual sentences with exactly the same length of the source sentence in a parallel sentence pair. After the first traversal of the entire parallel data set, we relax the length difference to 1. This process is repeated by relaxing the constraint until all the parallel data are paired with their own monolingual data.

2. A Prior Alignment Matrix

When \(L_{F_1}\) is removed, we can not obtain \(A'\) according to Algorithm 1 in the main paper which leads to the failure of calculating \(L_{F_2}\). Thus we propose a prior alignment to tackle this issue. For simplicity, we set \(n(\cdot)\) to be a copy function when doing the first \(XEnDec\), which means that we just randomly mask some words in the first round of \(XEnDec\). In the second \(XEnDec\), we want to combine \((x^p, y^p)\) and \((y^\circ, y)\). The alignment matrix \(A'\) for \((y^\circ, y)\) is constructed as follows.

If a word \(y_j\) in the target sentence \(y\) is picked in the source side which indicates \(y^\circ_j\) is picked and \(n_j = 0\), its attention value \(A'_{ji}\) if \(n_i = 0\) is assigned to \(\frac{p}{||i-m||}\), otherwise it is

\(\text{https://github.com/pytorch/fairseq/tree/master/examples/translation}\)
assigned to $\frac{1-p}{\|m\|_1}$ if $m_i = 1$. Conversely, If a word $y_j$ is not picked which indicates $m_j = 1$, its attention value $A'_{ji}$ is assigned to $\frac{p}{\|m\|_1}$ if $m_i = 0$, otherwise it is $\frac{1-p}{\|1-m\|_1}$ if $m_i = 1$.

References


