Plan, Attend, Generate: Planning for Sequence-to-Sequence Models

Francis Dutil*
University of Montreal (MILA)
frdutil@gmail.com

Caglar Gulcehre*
University of Montreal (MILA)
ca9lar@gmail.com

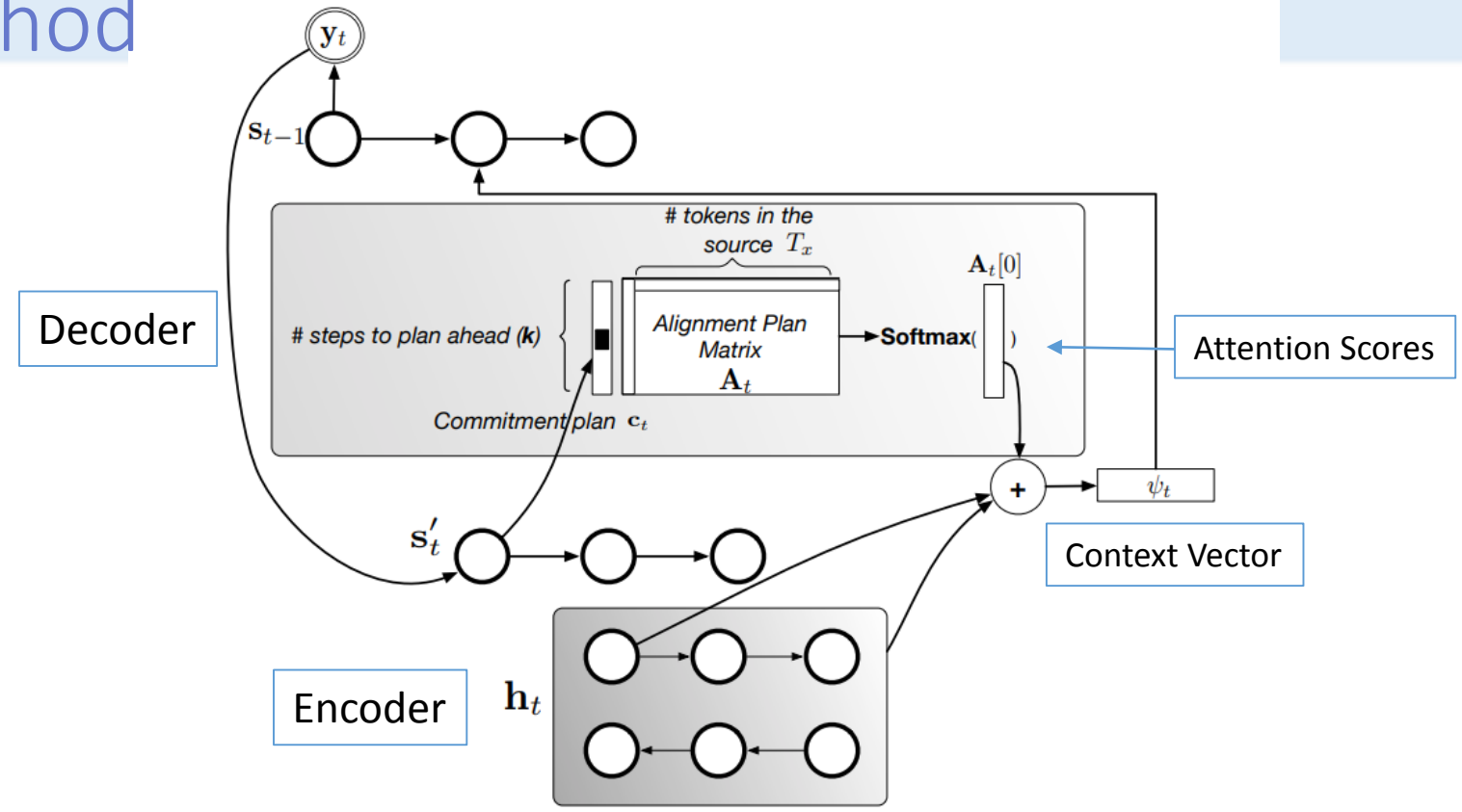
Adam Trischler
Microsoft Research Maluuba
adam.trischler@microsoft.com

Yoshua Bengio
University of Montreal (MILA)
yoshua.umontreal@gmail.com

Motivation

- Intuition: Natural sequences are not necessarily conceived and ordered according to only local, step-by-step interactions.
- Example: Sentences are not conceived one word at a time.
- Problem: Learning to generate long coherent sequences, or how to form alignments over long input contexts, is difficult for existing models.
- Planning is one way to induce coherence in sequential outputs like language.

Method



- We augment the decoder's internal state with (i) an alignment plan matrix *A* and (ii) a commitment plan vector *c*.
- $\mathbf{A}_t \in \mathbb{R}^{k \times |X|}$ is a sequence of probability distributions over input tokens, for next k time-steps.
- $\mathbf{c}_t \in \mathbb{R}^k$ governs whether to follow the alignment plan at the current step or to recompute it.

Update the Plan

- We denote by g_t the first element of the discretized commitment plan $\bar{c}_t$, i.e., $g_t = \bar{c}_t[0]$, which is a binary indicator. ($\bar{\mathbf{c}}_t = \texttt{one\_hot}(\mathbf{c}_t)$.)
- The model only updates its alignment plan when the current commitment switch g_t is active. Otherwise it uses the alignments planned and committed at previous time-steps

Algorithm 1: Pseudocode for updating the alignment plan and commitment vector.

for $j \in \{1, \cdots |X|\}$ **do**
  for $t \in \{1, \cdots |Y|\}$ **do**
    if $g_t = 1$ **then**
      $\mathbf{c}_t = \text{softmax}(f_c(\mathbf{s}_{t-1}))$
      $\beta_t^j = f_r(\mathbf{A}_{t-1}[j])$ {Read alignment plan}
      $\bar{\mathbf{A}}_t[i] = f_{\text{align}}(\mathbf{s}_{t-1}, \mathbf{h}_j, \beta_t^j, \mathbf{y}_t)$ {Compute candidate alignment plan}
      $\mathbf{u}_{tj} = f_{\text{up}}(\mathbf{h}_j, \mathbf{s}_{t-1}, \psi_{t-1})$ {Compute update gate}
      $\mathbf{A}_t = (1 - \mathbf{u}_{tj}) \odot \mathbf{A}_{t-1} + \mathbf{u}_{tj} \odot \bar{\mathbf{A}}_t$ {Update alignment plan}
    else
      $\mathbf{A}_t = \rho(\mathbf{A}_{t-1})$ {Shift alignment plan}
      $\mathbf{c}_t = \rho(\mathbf{c}_{t-1})$ {Shift commitment plan}
    end if
    Compute the alignment as $\alpha_t = \texttt{softmax}(\mathbf{A}_t[0])$
  end for
end for

An Alternative Method (rPAG)

- Reuse the alignment vector from the previous time-step until the commitment switch activates.
- Learn an explicit segmentation with an implicit planning mechanism in an unsupervised fashion.
- Reduce the computational complexity and memory consumption.

Algorithm 2: Pseudocode for updating the repeat alignment and commitment vector.

for $j \in \{1, \cdots |X|\}$ **do**
  for $t \in \{1, \cdots |Y|\}$ **do**
    if $g_t = 1$ **then**
      $\mathbf{c}_t = \texttt{softmax}(f_c(\mathbf{s}_{t-1}, \psi_{t-1}))$
      $\alpha_t = \texttt{softmax}(f_{\text{align}}(\mathbf{s}_{t-1}, \mathbf{h}_j, \mathbf{y}_t))$
    else
      $\mathbf{c}_t = \rho(\mathbf{c}_{t-1})$ {Shift the commitment vector $\mathbf{c}_{t-1}$}
      $\alpha_t = \alpha_{t-1}$ {Reuse the old the alignment}
    end if
  end for
end for

Training

- Loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{n=1}^{N} \log p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}),$$

- The proposed model can learn to recompute very often, which decreases the utility of planning.
- Introduce a loss that penalizes the model for committing too often:

$$\mathcal{L}_{\text{com}} = \lambda_{\text{com}} \sum_{t=1}^{|X|} \sum_{i=0}^{k} \| \frac{1}{k} - \mathbf{c}_{ti} \|_2^2,$$

Experiments

- Baseline: encoder-decoder(two layers) architecture with attention.
- Character-level Neural Machine Translation:
  - ➢ learn alignments for long sequences is difficult for existing models.
  - ➢ a model often must learn to segment them correctly.

	Model	Layer Norm	Dev	Test 2014	Test 2015
En→De	Baseline	✗	21.57	21.33	**23.45**
	Baseline†	✗	21.4	21.16	22.1
	Baseline†	✓	21.65	21.69	22.55
	PAG	✗	21.92	21.93	22.42
		✓	**22.44**	**22.59**	**23.18**
	rPAG	✗	21.98	22.17	22.85
		✓	22.33	22.35	22.83
En→Cs	Baseline	✗	17.68	19.27	16.98
	Baseline†	✓	19.1	21.35	18.79
	PAG	✗	18.9	20.6	18.88
		✓	**19.44**	**21.64**	**19.48**
	rPAG	✗	18.66	21.18	19.14
En→Fi	Baseline	✗	11.19	-	10.93
	Baseline†	✓	11.26	-	10.71
	PAG	✗	12.09	-	11.08
		✓	**12.85**	-	**12.15**
	rPAG	✗	11.76	-	11.02

Table 1: The results of different models on the WMT'15 tasks for English to German, English to Czech, and English to Finnish language pairs. We report BLEU scores of each model computed via the *multi-blue.perl* script. The best-score of each model for each language pair appears in bold-face. We use *newstest2013* as our development set, *newstest2014* as our "Test 2014" and *newstest2015* as our "Test 2015" set. (†) denotes the results of the baseline that we trained using the hyperparameters reported in Chung et al. (2016) and the code provided with that paper. For our baseline, we only report the median result, and do not have multiple runs of our models. On WMT'14 and WMT'15 for En$rightarrow$De character-level NMT, Kalchbrenner et al. (2016) have reported better results with deeper auto-regressive convolutional models (Bytenets), 23.75 and 26.26 respectively.

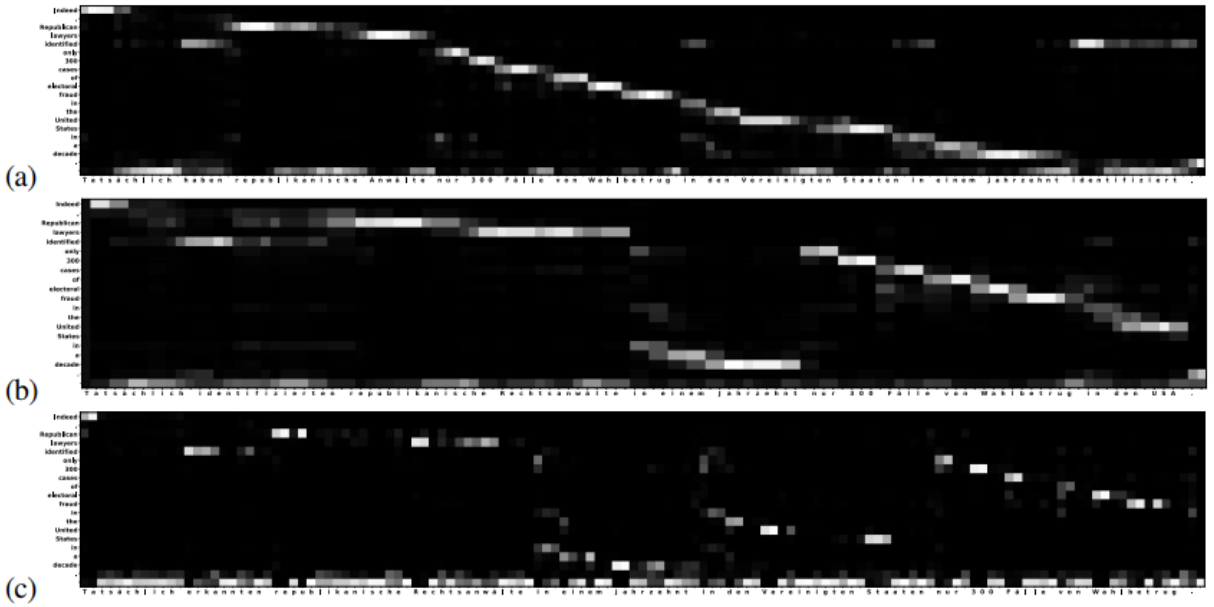


Figure 2: We visualize the alignments learned by PAG in (a), rPAG in (b), and our baseline model with a 2-layer GRU decoder using $\mathbf{h}_2$ for the attention in (c). As depicted, the alignments learned by PAG and rPAG are smoother than those of the baseline. The baseline tends to put too much attention on the last token of the sequence, defaulting to this empty location in alternation with more relevant locations. Our model, however, places higher weight on the last token usually when no other good alignments exist. We observe that rPAG tends to generate less monotonic alignments in general.

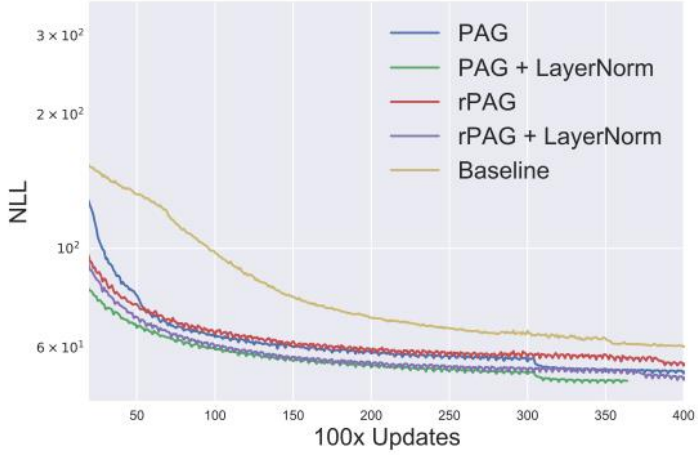


Figure 4: Learning curves for different models on WMT'15 for En→De. Models with the planning mechanism converge faster than our baseline (which has larger capacity).