Token-level and sequence-level loss smoothing for RNN language models

Maha Elbayad[1,2] **Laurent Besacier**[1] **Jakob Verbeek**[2]
Univ. Grenoble Alpes, CNRS, Grenoble INP, Inria, LIG, LJK, F-38000 Grenoble France
[1] firstname.lastname@univ-grenoble-alpes.fr
[2] firstname.lastname@inria.fr

Abstract

Despite the effectiveness of recurrent neural network language models, their maximum likelihood estimation suffers from two limitations. It treats all sentences that do not match the ground truth as equally poor, ignoring the structure of the output space. Second, it suffers from "exposure bias": during training tokens are predicted given ground-truth sequences, while at test time prediction is conditioned on generated output sequences. To overcome these limitations we build upon the recent reward augmented maximum likelihood approach *i.e.* sequence-level smoothing that encourages the model to predict sentences close to the ground truth according to a given performance metric. We extend this approach to token-level loss smoothing, and propose improvements to the sequence-level smoothing approach. Our experiments on two different tasks, image captioning and machine translation, show that token-level and sequence-level loss smoothing are complementary, and significantly improve results.

1 Introduction

Recurrent neural networks (RNNs) have recently proven to be very effective sequence modeling tools, and are now state of the art for tasks such as machine translation (Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015), image captioning (Kiros et al., 2014; Vinyals et al., 2015; Anderson et al., 2017) and automatic speech recognition (Chorowski et al., 2015; Chiu et al., 2017).

The basic principle of RNNs is to iteratively compute a vectorial sequence representation, by applying at each time-step the same trainable function to compute the new network state from the previous state and the last symbol in the sequence. These models are typically trained by maximizing the likelihood of the target sentence given an encoded source (text, image, speech).

Maximum likelihood estimation (MLE), however, has two main limitations. First, the training signal only differentiates the ground-truth target output from all other outputs. It treats all other output sequences as equally incorrect, regardless of their semantic proximity from the ground-truth target. While such a "zero-one" loss is probably acceptable for coarse grained classification of images, *e.g.* across a limited number of basic object categories (Everingham et al., 2010) it becomes problematic as the output space becomes larger and some of its elements become semantically similar to each other. This is in particular the case for tasks that involve natural language generation (captioning, translation, speech recognition) where the number of possible outputs is practically unbounded. For natural language generation tasks, evaluation measures typically do take into account structural similarity, *e.g.* based on n-grams, but such structural information is not reflected in the MLE criterion. The second limitation of MLE is that training is based on predicting the next token given the input and preceding *ground-truth* output tokens, while at test time the model predicts conditioned on the input and the so-far *generated* output sequence. Given the exponentially large output space of natural language sentences, it is not obvious that the learned RNNs generalize well beyond the relatively sparse distribution of ground-truth sequences used during MLE optimization. This phenomenon is known as "exposure bias" (Ranzato et al., 2016; Bengio et al., 2015).

MLE minimizes the KL divergence between a target Dirac distribution on the ground-truth sentence(s) and the model's distribution. In this pa-

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 2094–2103
Melbourne, Australia, July 15 - 20, 2018.

per, we build upon the "loss smoothing" approach by Norouzi et al. (2016), which smooths the Dirac target distribution over similar sentences, increasing the support of the training data in the output space. We make the following main contributions:

- We propose a token-level loss smoothing approach, using word-embeddings, to achieve smoothing among semantically similar terms, and we introduce a special procedure to promote rare tokens.
- For sequence-level smoothing, we propose to use restricted token replacement vocabularies, and a "lazy evaluation" method that significantly speeds up training.
- We experimentally validate our approach on the MSCOCO image captioning task and the WMT'14 English to French machine translation task, showing that on both tasks combining token-level and sequence-level loss smoothing improves results significantly over maximum likelihood baselines.

In the remainder of the paper, we review the existing methods to improve RNN training in Section 2. Then, we present our token-level and sequence-level approaches in Section 3. Experimental evaluation results based on image captioning and machine translation tasks are laid out in Section 4.

2 Related work

Previous work aiming to improve the generalization performance of RNNs can be roughly divided into three categories: those based on regularization, data augmentation, and alternatives to maximum likelihood estimation.

Regularization techniques are used to increase the smoothness of the function learned by the network, *e.g.* by imposing an ℓ_2 penalty on the network weights, also known as "weight decay". More recent approaches mask network activations during training, as in dropout (Srivastava et al., 2014) and its variants adapted to recurrent models (Pham et al., 2014; Krueger et al., 2017). Instead of masking, batch-normalization (Ioffe and Szegedy, 2015) rescales the network activations to avoid saturating the network's non-linearities. Instead of regularizing the network parameters or activations, it is also possible to directly regularize based on the entropy of the output distribution (Pereyra et al., 2017).

Data augmentation techniques improve the robustness of the learned models by applying transformations that might be encountered at test time to the training data. In computer vision, this is common practice, and implemented by, *e.g.*, scaling, cropping, and rotating training images (LeCun et al., 1998; Krizhevsky et al., 2012; Paulin et al., 2014). In natural language processing, examples of data augmentation include input noising by randomly dropping some input tokens (Iyyer et al., 2015; Bowman et al., 2015; Kumar et al., 2016), and randomly replacing words with substitutes sampled from the model (Bengio et al., 2015). Xie et al. (2017) introduced data augmentation schemes for RNN language models that leverage n-gram statistics in order to mimic Kneser-Ney smoothing of n-grams models. In the context of machine translation, Fadaee et al. (2017) modify sentences by replacing words with rare ones when this is plausible according to a pre-trained language model, and substitutes its equivalent in the target sentence using automatic word alignments. This approach, however, relies on the availability of additional monolingual data for language model training.

The *de facto* standard way to train RNN language models is maximum likelihood estimation (MLE) (Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015). The sequential factorization of the sequence likelihood generates an additive structure in the loss, with one term corresponding to the prediction of each output token given the input and the preceding ground-truth output tokens. In order to directly optimize for sequence-level structured loss functions, such as measures based on n-grams like BLEU or CIDER, Ranzato et al. (2016) use reinforcement learning techniques that optimize the expectation of a sequence-level reward. In order to avoid early convergence to poor local optima, they pre-train the model using MLE.

Leblond et al. (2018) build on the learning to search approach to structured prediction (Daumé III et al., 2009; Chang et al., 2015) and adapts it to RNN training. The model generates candidate sequences at each time-step using all possible tokens, and scores these at sequence-level to derive a training signal for each time step. This leads to an approach that is structurally close to MLE, but computationally expensive. Norouzi et al. (2016) introduce a reward augmented maximum likelihood (RAML) approach, that incorpo-

rates a notion of sequence-level reward without facing the difficulties of reinforcement learning. They define a target distribution over output sentences using a soft-max over the reward over all possible outputs. Then, they minimize the KL divergence between the target distribution and the model's output distribution. Training with a general reward distribution is similar to MLE training, except that we use multiple sentences sampled from the target distribution instead of only the ground-truth sentences.

In our work, we build upon the work of Norouzi et al. (2016) by proposing improvements to sequence-level smoothing, and extending it to token-level smoothing. Our token-level smoothing approach is related to the label smoothing approach of Szegedy et al. (2016) for image classification. Instead of maximizing the probability of the correct class, they train the model to predict the correct class with a large probability and all other classes with a small uniform probability. This regularizes the model by preventing over-confident predictions. In natural language generation with large vocabularies, preventing such "narrow" over-confident distributions is imperative, since for many tokens there are nearly interchangeable alternatives.

3 Loss smoothing for RNN training

We briefly recall standard recurrent neural network training, before presenting sequence-level and token-level loss smoothing below.

3.1 Maximum likelihood RNN training

We are interested in modeling the conditional probability of a sequence $y = (y_1, \dots, y_T)$ given a conditioning observation x,

$$p_\theta(y|x) = \prod_{t=1}^{T} p_\theta(y_t|x, y_{<t}), \tag{1}$$

where $y_{<t} = (y_1, \dots, y_{t-1})$, the model parameters are given by θ, and x is a source sentence or an image in the contexts of machine translation and image captioning, respectively.

In a recurrent neural network, the sequence y is predicted based on a sequence of states h_t,

$$p_\theta(y_t|x, y_{<t}) = p_\theta(y_t|h_t), \tag{2}$$

where the RNN state is computed recursively as

$$h_t = \begin{cases} f_\theta(h_{t-1}, y_{t-1}, x) & \text{for } t \in \{1, ..T\}, \\ g_\theta(x) & \text{for } t = 0. \end{cases} \tag{3}$$

The input is encoded by g_θ and used to initialize the state sequence, and f_θ is a non-linear function that updates the state given the previous state h_{t-1}, the last output token y_{t-1}, and possibly the input x. The state update function can take different forms, the ones including gating mechanisms such as LSTMs (Hochreiter and Schmidhuber, 1997) and GRUs (Chung et al., 2014) are particularly effective to model long sequences.

In standard teacher-forced training, the hidden states will be computed by forwarding the ground truth sequence y^* *i.e.* in Eq. (3), the RNN has access to the true previous token y^*_{t-1}. In this case we will note the hidden states h^*_t.

Given a ground-truth target sequence y^*, maximum likelihood estimation (MLE) of the network parameters θ amounts to minimizing the loss

$$\ell_{\text{MLE}}(y^*, x) = -\ln p_\theta(y^*|x) \tag{4}$$

$$= -\sum_{t=1}^{T} \ln p_\theta(y^*_t|h^*_t). \tag{5}$$

The loss can equivalently be expressed as the KL-divergence between a Dirac centered on the target output (with $\delta_a(x) = 1$ at $x = a$ and 0 otherwise) and the model distribution, either at the sequence-level or at the token-level:

$$\ell_{\text{MLE}}(y^*, x) = D_{\text{KL}}\big(\delta_{y^*} || p_\theta(y|x)\big) \tag{6}$$

$$= \sum_{t=1}^{T} D_{\text{KL}}\big(\delta_{y^*_t} || p_\theta(y_t|h^*_t)\big). \tag{7}$$

Loss smoothing approaches considered in this paper consist in replacing the Dirac on the ground-truth sequence with distributions with larger support. These distributions can be designed in such a manner that they reflect which deviations from ground-truth predictions are preferred over others.

3.2 Sequence-level loss smoothing

The reward augmented maximum likelihood approach of Norouzi et al. (2016) consists in replacing the sequence-level Dirac δ_{y^*} in Eq. (6) with a distribution

$$r(y|y^*) \propto \exp r(y, y^*)/\tau, \tag{8}$$

where $r(y, y^*)$ is a "reward" function that measures the quality of sequence y w.r.t. y^*, *e.g.* metrics used for evaluation of natural language processing tasks can be used, such as BLEU (Papineni et al., 2002) or CIDEr (Vedantam et al.,

2015). The temperature parameter τ controls the concentration of the distribution around y^*. When $m > 1$ ground-truth sequences are paired with the same input x, the reward function can be adapted to fit this setting and be defined as $r(y, \{y^{*(1)}, \ldots, y^{*(m)}\})$. The sequence-level smoothed loss function is then given by

$$\begin{aligned}\ell_{\text{Seq}}(y^*, x) &= D_{\text{KL}}\big(r(y|y^*)||p_\theta(y|x)\big) \\ &= H(r(y|y^*)) - \mathbb{E}_r[\ln p_\theta(y|x)]\,, \quad (9)\end{aligned}$$

where the entropy term $H(r(y|y^*))$ does not depend on the model parameters θ.

In general, expectation in Eq. (9) is intractable due to the exponentially large output space, and replaced with a Monte-Carlo approximation:

$$\mathbb{E}_r[-\ln p_\theta(y|x)] \approx -\sum_{l=1}^{L} \ln p_\theta(y^l|x). \quad (10)$$

Stratified sampling. Norouzi et al. (2016) show that when using the Hamming or edit distance as a reward, we can sample directly from $r(y|y^*)$ using a stratified sampling approach. In this case sampling proceeds in three stages. (i) Sample a distance d from $\{0, \ldots, T\}$ from a prior distribution on d. (ii) Uniformly select d positions in the sequence to be modified. (iii) Sample the d substitutions uniformly from the token vocabulary.

Details on the construction of the prior distribution on d for a reward based on the Hamming distance can be found in Appendix A.

Importance sampling. For a reward based on BLEU or CIDEr , we cannot directly sample from $r(y|y^*)$ since the normalizing constant, or "partition function", of the distribution is intractable to compute. In this case we can resort to importance sampling. We first sample L sequences y^l from a tractable proposal distribution $q(y|y^*)$. We then compute the importance weights

$$\omega_l \approx \frac{r(y^l|y^*)/q(y^l|y^*)}{\sum_{k=1}^{L} r(y^k|y^*)/q(y^k|y^*)}, \quad (11)$$

where $r(y^k|y^*)$ is the un-normalized reward distribution in Eq. (8). We finally approximate the expectation by reweighing the samples in the Monte Carlo approximation as

$$\mathbb{E}_r[-\ln p_\theta(y|x)] \approx -\sum_{l=1}^{L} \omega_l \ln p_\theta(y^l|x). \quad (12)$$

In our experiments we use a proposal distribution based on the Hamming distance, which allows for tractable stratified sampling, and generates sentences that do not stray away from the ground truth.

We propose two modifications to the sequence-level loss smoothing of Norouzi et al. (2016): sampling to a restricted vocabulary (described in the following paragraph) and lazy sequence-level smoothing (described in section 3.4).

Restricted vocabulary sampling. In the stratified sampling method for Hamming and edit distance rewards, instead of drawing from the large vocabulary $\mathcal{V}$, containing typically in the order of 10^4 words or more, we can restrict ourselves to a smaller subset $\mathcal{V}_{sub}$ more adapted to our task. We considered three different possibilities for $\mathcal{V}_{sub}$.

$\mathcal{V}$: the full vocabulary from which we sample uniformly (default), or draw from our token-level smoothing distribution defined below in Eq. (13).

$\mathcal{V}_{refs}$: uniformly sample from the set of tokens that appear in the ground-truth sentence(s) associated with the current input.

$\mathcal{V}_{batch}$: uniformly sample from the tokens that appear in the ground-truth sentences across all inputs that appear in a given training mini-batch.

Uniformly sampling from $\mathcal{V}_{batch}$ has the effect of boosting the frequencies of words that appear in many reference sentences, and thus approximates to some extent sampling substitutions from the uni-gram statistics of the training set.

3.3 Token-level loss smoothing

While the sequence-level smoothing can be directly based on performance measures of interest such as BLEU or CIDEr, the support of the smoothed distribution is limited to the number of samples drawn during training. We propose smoothing the token-level Diracs $\delta_{y_t^*}$ in Eq. (7) to increase its support to similar tokens. Since we apply smoothing to each of the tokens independently, this approach implicitly increases the support to an exponential number of sequences, unlike the sequence-level smoothing approach. This comes at the price, however, of a naive token-level independence assumption in the smoothing.

We define the smoothed token-level distribution, similar as the sequence-level one, as a softmax over a token-level "reward" function,

$$r(y_t|y_t^*) \propto \exp r(y_t, y_t^*)/\tau, \quad (13)$$

where τ is again a temperature parameter. As a token-level reward $r(y_t, y_t^*)$ we use the cosine similarity between y_t and y_t^* in a semantic word-embedding space. In our experiments we use GloVe (Pennington et al., 2014); preliminary experiments with word2vec (Mikolov et al., 2013) yielded somewhat worse results.

Promoting rare tokens. We can further improve the token-level smoothing by promoting rare tokens. To do so, we penalize frequent tokens when smoothing over the vocabulary, by subtracting $\beta \operatorname{freq}(y_t)$ from the reward, where $\operatorname{freq}(\cdot)$ denotes the term frequency and β is a non-negative weight. This modification encourages frequent tokens into considering the rare ones. We experimentally found that it is also beneficial for rare tokens to boost frequent ones, as they tend to have mostly rare tokens as neighbors in the word-embedding space. With this in mind, we define a new token-level reward as:

$$r^{\text{freq}}(y_t, y_t^*) = r(y_t, y_t^*) - \beta \min\left(\frac{\operatorname{freq}(y_t)}{\operatorname{freq}(y_t^*)}, \frac{\operatorname{freq}(y_t^*)}{\operatorname{freq}(y_t)}\right), \tag{14}$$

where the penalty term is strongest if both tokens have similar frequencies.

3.4 Combining losses

In both loss smoothing methods presented above, the temperature parameter τ controls the concentration of the distribution. As τ gets smaller the distribution peaks around the ground-truth, while for large τ the uniform distribution is approached. We can, however, not separately control the spread of the distribution and the mass reserved for the ground-truth output. We therefore introduce a second parameter $\alpha \in [0, 1]$ to interpolate between the Dirac on the ground-truth and the smooth distribution. Using $\bar{\alpha} = 1 - \alpha$, the sequence-level and token-level loss functions are then defined as

$$\begin{aligned}\ell^{\alpha}_{\text{Seq}}(y^*, x) &= \alpha \ell_{\text{Seq}}(y^*, x) + \bar{\alpha} \ell_{\text{MLE}}(y^*, x) \quad (15)\\ &= \alpha \mathbb{E}_r[\ell_{\text{MLE}}(y, x)] + \bar{\alpha} \ell_{\text{MLE}}(y^*, x)\\ \ell^{\alpha}_{\text{Tok}}(y^*, x) &= \alpha \ell_{\text{Tok}}(y^*, x) + \bar{\alpha} \ell_{\text{MLE}}(y^*, x) \quad (16)\end{aligned}$$

To benefit from both sequence-level and token-level loss smoothing, we also combine them by applying token-level smoothing to the different sequences sampled for the sequence-level smoothing. We introduce two mixing parameters α_1 and α_2. The first controls to what extent sequence-level smoothing is used, while the second controls to what extent token-level smoothing is used. The combined loss is defined as

$$\begin{aligned}\ell^{\alpha_1,\alpha_2}_{\text{Seq, Tok}}(y^*, x, r) &= \alpha_1 \mathbb{E}_r[\ell_{\text{Tok}}(y, x)] + \bar{\alpha}_1 \ell_{\text{Tok}}(y^*, x)\\ &= \alpha_1 \mathbb{E}_r[\alpha_2 \ell_{\text{Tok}}(y, x) + \bar{\alpha}_2 \ell_{\text{MLE}}(y, x)]\\ &\quad + \bar{\alpha}_1(\alpha_2 \ell_{\text{Tok}}(y^*, x) + \bar{\alpha}_2 \ell_{\text{MLE}}(y^*, x)).\end{aligned} \tag{17}$$

In our experiments, we use held out validation data to set mixing and temperature parameters.

Algorithm 1 Sequence-level smoothing algorithm

Input: x, y^*
Output: $\ell^{\alpha}_{\text{seq}}(x, y^*)$
  Encode x to initialize the RNN
  Forward y^* in the RNN to compute the hidden states h_t^*
  Compute the MLE loss $\ell_{\text{MLE}}(y^*, x)$
  for $l \in \{1, \dots, L\}$ **do**
    Sample $y^l \sim r(\dot{\mid} y^*)$
    if Lazy **then**
      Compute $\ell(y^l, x) = -\sum_t \log p_\theta(y_t^l | h_t^*)$
    else
      Forward y^l in the RNN to get its hidden states h_t^l
      Compute $\ell(y^l, x) = \ell_{\text{MLE}}(y^l, x)$
    end if
  end for
  $\ell^{\alpha}_{\text{Seq}}(x, y^*) = \bar{\alpha} \ell_{\text{MLE}}(y^*, x) + \frac{\alpha}{L} \sum_l \ell(y^l, x)$

Lazy sequence smoothing. Although sequence-level smoothing is computationally efficient compared to reinforcement learning approaches (Ranzato et al., 2016; Rennie et al., 2017), it is slower compared to MLE. In particular, we need to forward each of the samples y^l through the RNN in teacher-forcing mode so as to compute its hidden states h_t^l, which are used to compute the sequence MLE loss as

$$\ell_{\text{MLE}}(y^l, x) = -\sum_{t=1}^{T} \ln p_\theta(y_t^l | h_t^l). \tag{18}$$

To speed up training, and since we already forward the ground truth sequence in the RNN to evaluate the MLE part of $\ell^{\alpha}_{\text{Seq}}(y^*, x)$, we propose to use the same hidden states h_t^* to compute both the MLE and the sequence-level smoothed loss. In this case:

$$\ell_{\text{lazy}}(y^l, x) = -\sum_{t=1}^{T} \ln p_\theta(y_t^l | h_t^*) \tag{19}$$

In this manner, we only have a single instead of $L + 1$ forwards-passes in the RNN. We provide the pseudo-code for training in Algorithm 1.

			Without attention			With attention		
Loss	Reward	$\mathcal{V}_{sub}$	BLEU-1	BLEU-4	CIDEr	BLEU-1	BLEU-4	CIDEr
MLE			70.63	30.14	93.59	73.40	33.11	101.63
MLE + γH			70.79	30.29	93.61	72.68	32.15	99.77
Tok	Glove sim		71.94	31.27	95.79	73.49	32.93	102.33
Tok	Glove sim r^{freq}		72.39	31.76	97.47	74.01	33.25	102.81
Seq	Hamming	$\mathcal{V}$	71.76	31.16	96.37	73.12	32.71	101.25
Seq	Hamming	$\mathcal{V}_{batch}$	71.46	31.15	**96.53**	73.26	**32.73**	101.90
Seq	Hamming	$\mathcal{V}_{refs}$	**71.80**	**31.63**	96.22	**73.53**	32.59	**102.33**
Seq, lazy	Hamming	$\mathcal{V}$	70.81	30.43	94.26	73.29	32.81	101.58
Seq, lazy	Hamming	$\mathcal{V}_{batch}$	71.85	31.13	**96.65**	73.43	32.95	**102.03**
Seq, lazy	Hamming	$\mathcal{V}_{refs}$	**71.96**	**31.23**	95.34	**73.53**	**33.09**	101.89
Seq	CIDEr	$\mathcal{V}$	71.05	30.46	94.40	73.08	32.51	101.84
Seq	CIDEr	$\mathcal{V}_{batch}$	71.51	31.17	95.78	**73.50**	**33.04**	**102.98**
Seq	CIDEr	$\mathcal{V}_{refs}$	**71.93**	**31.41**	**96.81**	73.42	32.91	102.23
Seq, lazy	CIDEr	$\mathcal{V}$	71.43	**31.18**	**96.32**	73.55	**33.19**	**102.94**
Seq, lazy	CIDEr	$\mathcal{V}_{batch}$	71.47	31.00	95.56	73.18	32.60	101.30
Seq, lazy	CIDEr	$\mathcal{V}_{refs}$	**71.82**	31.06	95.66	**73.92**	33.10	102.64
Tok-Seq	Hamming	$\mathcal{V}$	70.79	30.43	96.34	73.68	32.87	101.11
Tok-Seq	Hamming	$\mathcal{V}_{batch}$	72.28	31.65	96.73	73.86	33.32	102.90
Tok-Seq	Hamming	$\mathcal{V}_{refs}$	72.69	32.30	98.01	73.56	33.00	101.72
Tok-Seq	CIDEr	$\mathcal{V}$	70.80	30.55	96.89	73.31	32.40	100.33
Tok-Seq	CIDEr	$\mathcal{V}_{batch}$	72.13	31.71	96.92	73.61	32.67	101.41
Tok-Seq	CIDEr	$\mathcal{V}_{refs}$	**73.08**	**32.82**	**99.92**	**74.28**	**33.34**	**103.81**

Table 1: MS-COCO 's test set evaluation measures.

4 Experimental evaluation

In this section, we compare sequence prediction models trained with maximum likelihood (MLE) with our token and sequence-level loss smoothing on two different tasks: image captioning and machine translation.

4.1 Image captioning

4.1.1 Experimental setup.

We use the MS-COCO datatset (Lin et al., 2014), which consists of 82k training images each annotated with five captions. We use the standard splits of Karpathy and Li (2015), with 5k images for validation, and 5k for test. The test set results are generated via beam search (beam size 3) and are evaluated with the MS-COCO captioning evaluation tool. We report CIDEr and BLEU scores on this internal test set. We also report results obtained on the official MS-COCO server that additionally measures METEOR (Denkowski and Lavie, 2014) and ROUGE-L (Lin, 2004). We experiment with both non-attentive LSTMs (Vinyals et al., 2015) and the ResNet baseline of the state-of-the-art top-down attention (Anderson et al., 2017).

The MS-COCO vocabulary consists of 9,800 words that occur at least 5 times in the training set. Additional details and hyperparameters can be found in Appendix B.1.

4.1.2 Results and discussion

Restricted vocabulary sampling In this section, we evaluate the impact of the vocabulary subset from which we sample the modified sentences for sequence-level smoothing. We experiment with two rewards: CIDEr , which scores w.r.t. all five available reference sentences, and Hamming distance reward taking only a single reference into account. For each reward we train our (Seq) models with each of the three subsets detailed previously in Section 3.2, Restricted vocabulary sampling.

From the results in Table 1 we note that for the inattentive models, sampling from $\mathcal{V}_{refs}$ or $\mathcal{V}_{batch}$ has a better performance than sampling from the full vocabulary on all metrics. In fact, using these subsets introduces a useful bias to the model and improves performance. This improvement is most notable using the CIDEr reward that scores candidate sequences w.r.t. to multiple references, which stabilizes the scoring of the candidates.

With an attentive decoder, no matter the reward, re-sampling sentences with words from $\mathcal{V}_{ref}$ rather than the full vocabulary $\mathcal{V}$ is better for both reward functions, and all metrics. Additional experimental results, presented in Appendix B.2, obtained with a BLEU-4 reward, in its single and

	BLEU-1		BLEU-2		BLEU-3		BLEU-4		METEOR		ROUGE-L		CIDEr		SPICE	
	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40
Google NIC^{+} (Vinyals et al., 2015)	71.3	89.5	54.2	80.2	40.7	69.4	30.9	58.7	25.4	34.6	53.0	68.2	94.3	94.6	18.2	63.6
Hard-Attention (Xu et al., 2015)	70.5	88.1	52.8	77.9	38.3	65.8	27.7	53.7	24.1	32.2	51.6	65.4	86.5	89.3	17.2	59.8
ATT-FCN^{+} (You et al., 2016)	73.1	90.0	56.5	81.5	42.4	70.9	31.6	59.9	25.0	33.5	53.5	68.2	94.3	95.8	18.2	63.1
Review Net^{+} (Yang et al., 2016)	72.0	90.0	55.0	81.2	41.4	70.5	31.3	59.7	25.6	34.7	53.3	68.6	96.5	96.9	18.5	64.9
Adaptive^{+} (Lu et al., 2017)	74.8	92.0	58.4	84.5	44.4	74.4	33.6	63.7	26.4	35.9	55.0	70.5	104.2	105.9	19.7	67.3
SCST:Att2all$^{+\dagger}$ (Rennie et al., 2017)	78.1	93.7	61.9	86.0	47.0	75.9	35.2	64.5	27.0	35.5	56.3	70.7	114.7	116.7	-	-
LSTM-A3$^{+\dagger\circ}$ (Yao et al., 2017)	78.7	93.7	62.7	86.7	47.6	76.5	35.6	65.2	27.0	35.4	56.4	70.5	116	118	-	-
Up-Down$^{+\dagger\circ}$ (Anderson et al., 2017)	80.2	95.2	64.1	88.8	49.1	79.4	36.9	68.5	27.6	36.7	57.1	72.4	117.9	120.5	-	-
Ours: Tok-Seq CIDEr	72.6	89.7	55.7	80.9	41.2	69.8	30.2	58.3	25.5	34.0	53.5	68.0	96.4	99.4	-	-
Ours: Tok-Seq CIDEr $^{+}$	74.9	92.4	58.5	84.9	44.8	75.1	34.3	64.7	26.5	36.1	55.2	71.1	103.9	104.2	-	-

Table 2: MS-COCO 's server evaluation . ($^{+}$) for ensemble submissions, (†) for submissions with CIDEr optimization and ($^{\circ}$) for models using additional data.

multiple references variants, further corroborate this conclusion.

Lazy training. From the results of Table 1, we see that lazy sequence-level smoothing is competitive with exact non-lazy sequence-level smoothing, while requiring roughly equivalent training time as MLE. We provide detailed timing results in Appendix B.3.

Overall For reference, we include in Table 1 baseline results obtained using MLE, and our implementation of MLE with entropy regularization (MLE+γH) (Pereyra et al., 2017), as well as the RAML approach of Norouzi et al. (2016) which corresponds to sequence-level smoothing based on the Hamming reward and sampling replacements from the full vocabulary (Seq, Hamming, $\mathcal{V}$)

We observe that entropy smoothing is not able to improve performance much over MLE for the model without attention, and even deteriorates for the attention model. We improve upon RAML by choosing an adequate subset of vocabulary for substitutions.

We also report the performances of token-level smoothing, where the promotion of rare tokens boosted the scores in both attentive and non-attentive models.

For sequence-level smoothing, choosing a task-relevant reward with importance sampling yielded better results than plain Hamming distance.

Moreover, we used the two smoothing schemes (Tok-Seq) and achieved the best results with CIDEr as a reward for sequence-level smoothing combined with a token-level smoothing that promotes rare tokens improving CIDEr from 93.59 (MLE) to 99.92 for the model without attention, and improving from 101.63 to 103.81 with attention.

Qualitative results. In Figure 1 we showcase captions obtained with MLE and our three variants of smoothing *i.e.* token-level (Tok), sequence-level (Seq) and the combination (Tok-Seq). We note that the sequence-level smoothing tend to generate lengthy captions overall, which is maintained in the combination. On the other hand, the token-level smoothing allows for a better recognition of objects in the image that stems from the robust training of the classifier *e.g.* the 'cement block' in the top right image or the carrots in the bottom right. More examples are available in Appendix B.4

Comparison to the state of the art. We compare our model to state-of-the-art systems on the MS-COCO evaluation server in Table 2. We submitted a single model (Tok-Seq, CIDEr , $\mathcal{V}_{refs}$) as well as an ensemble of five models with different initializations trained on the training set plus 35k images from the dev set (a total of 117k images) to the MS-COCO server. The three best results on the server (Rennie et al., 2017; Yao et al., 2017; Anderson et al., 2017) are trained in two stages where they first train using MLE, before switching to policy gradient methods based on CIDEr. Anderson et al. (2017) reported an increase of 5.8% of CIDEr on the test split after the CIDEr optimization. Moreover, Yao et al. (2017) uses additional information about image regions to train the attributes classifiers, while Anderson et al. (2017) pre-trains its bottom-up attention model on the Visual Genome dataset (Krishna et al., 2017). Lu et al. (2017); Yao et al. (2017) use the same CNN encoder as ours (ResNet-152), (Vinyals et al., 2015; Yang et al., 2016) use Inception-v3 (Szegedy et al., 2016) for image encoding and Rennie et al. (2017); Anderson et al.

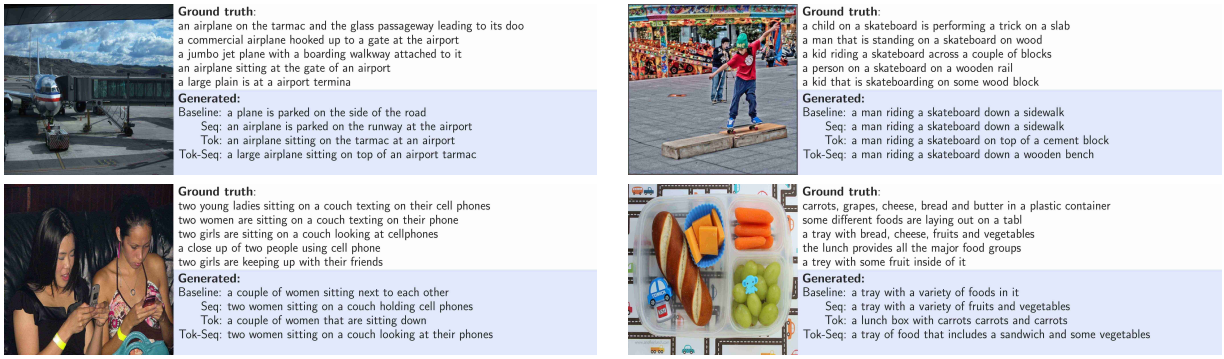

Figure 1: Examples of generated captions with the baseline MLE and our models with attention.

(2017) use Resnet-101, both of which have similar performances to ResNet-152 on ImageNet classification (Canziani et al., 2016).

4.2 Machine translation

4.2.1 Experimental setup.

For this task we validate the effectiveness of our approaches on two different datasets. The first is WMT'14 English to French, in its filtered version, with 12M sentence pairs obtained after dynamically selecting a "clean" subset of 348M words out of the original "noisy" 850M words (Bahdanau et al., 2015; Cho et al., 2014; Sutskever et al., 2014). The second benchmark is IWSLT'14 German to English consisting of around 150k pairs for training. In all our experiments we use the attentive model of (Bahdanau et al., 2015) The hyperparameters of each of these models as well as any additional pre-processing can be found in Appendix C.1

To assess the translation quality we report the BLEU-4 metric.

4.2.2 Results and analysis

Loss	Reward	$\mathcal{V}_{sub}$	WMT'14	IWSLT'14
MLE			30.03	27.55
tok	Glove sim		30.16	27.69
tok	Glove sim r^{freq}		30.19	27.83
Seq	Hamming	$\mathcal{V}$	30.85	27.98
Seq	Hamming	$\mathcal{V}_{batch}$	31.18	28.54
Seq	BLEU-4	$\mathcal{V}_{batch}$	31.29	28.56
Tok-Seq	Hamming	$\mathcal{V}_{batch}$	31.36	28.70
Tok-Seq	BLEU-4	$\mathcal{V}_{batch}$	31.39	28.74

Table 3: Tokenized BLEU score on WMT'14 En-Fr evaluated on the news-test-2014 set. And Tokenzied, case-insensitive BLEU on IWSLT'14 De-En.

We present our results in Table 3. On both benchmarks, we improve on both MLE and RAML approach of Norouzi et al. (2016) (Seq, Hamming, $\mathcal{V}$): using the smaller batch-vocabulary for replacement improves results, and using importance sampling based on BLEU-4 further boosts results. In this case, unlike in the captioning experiment, token-level smoothing brings smaller improvements. The combination of both smoothing approaches gives best results, similar to what was observed for image captioning, improving the MLE BLEU-4 from 30.03 to 31.39 on WMT'14 and from 27.55 to 28.74 on IWSLT'14. The outputs of our best model are compared to the MLE in some examples showcased in Appendix C.

5 Conclusion

We investigated the use of loss smoothing approaches to improve over maximum likelihood estimation of RNN language models. We generalized the sequence-level smoothing RAML approach of Norouzi et al. (2016) to the token-level by smoothing the ground-truth target across semantically similar tokens. For the sequence-level, which is computationally expensive, we introduced an efficient "lazy" evaluation scheme, and introduced an improved re-sampling strategy. Experimental evaluation on image captioning and machine translation demonstrates the complementarity of sequence-level and token-level loss smoothing, improving over both the maximum likelihood and RAML.

Acknowledgment. This work has been partially supported by the grant ANR-16-CE23-0006 "Deep in France" and LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01).

References

P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. 2017. Bottom-up and top-down attention for image captioning and visual question answering. *arXiv preprint arXiv:1707.07998*.

D. Bahdanau, K. Cho, and Y. Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *NIPS*.

S. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio. 2015. Generating sentences from a continuous space. In *CoNLL*.

A. Canziani, A. Paszke, and E. Culurciello. 2016. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*.

K.-W. Chang, A. Krishnamurthy, A. Agarwal, H. Daumé III, and J. Langford. 2015. Learning to search better than your teacher. In *ICML*.

C.-C. Chiu, T. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R.-J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, and M. Bacchiani. 2017. State-of-the-art speech recognition with sequence-to-sequence models. *arXiv preprint arXiv:1712.01769*.

K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Empirical Methods in Natural Language Processing*.

J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio. 2015. Attention-based models for speech recognition. In *NIPS*.

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS Deep Learning Workshop*.

H. Daumé III, J. Langford, and D. Marcu. 2009. Search-based structured prediction. *Machine Learning*, 75(3):297–325.

M. Denkowski and A. Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Workshop on statistical machine translation*.

M. Everingham, L. van Gool, C. Williams, J. Winn, and A. Zisserman. 2010. The pascal visual object classes (VOC) challenge. *IJCV*, 88(2):303–338.

M. Fadaee, A. Bisazza, and C. Monz. 2017. Data augmentation for low-resource neural machine translation. In *ACL*.

K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

S. Ioffe and C. Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*.

M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *ACL*.

A. Karpathy and Fei-Fei Li. 2015. Deep visual-semantic alignments for generating image descriptions. In *CVPR*.

D. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

R. Kiros, R. Salakhutdinov, and R. Zemel. 2014. Multimodal neural language models. In *ICML*.

R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. Shamma, M. Bernstein, and L. Fei-Fei. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 123(1):32–73.

A. Krizhevsky, I. Sutskever, and G. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

D. Krueger, T. Maharaj, J. Kramár, M. Pezeshki, N. Ballas, N. Ke, A. Goyal, Y. Bengio, H. Larochelle, A. Courville, and C. Pal. 2017. Zoneout: Regularizing RNNs by randomly preserving hidden activations. In *ICLR*.

A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *ICML*.

R. Leblond, J.-B. Alayrac, A. Osokin, and S. Lacoste-Julien. 2018. SeaRnn: Training RNNs with global-local losses. In *ICLR*.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, pages 2278–2324.

C.-Y. Lin. 2004. Rouge: a package for automatic evaluation of summaries. In *ACL Workshop Text Summarization Branches Out*.

T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick. 2014. Microsoft COCO: common objects in context. In *ECCV*.

J. Lu, C. Xiong, D. Parikh, and R. Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *CVPR*.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

M. Norouzi, S. Bengio, Z. Chen, N. Jaitly, M. Schuster, Y. Wu, and D. Schuurmans. 2016. Reward augmented maximum likelihood for neural structured prediction. In *NIPS*.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

M. Paulin, J. Revaud, Z. Harchaoui, F. Perronnin, and C. Schmid. 2014. Transformation pursuit for image classification. In *CVPR*.

M. Pedersoli, T. Lucas, C. Schmid, and J. Verbeek. 2017. Areas of attention for image captioning. In *ICCV*.

J. Pennington, R. Socher, and C. Manning. 2014. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*.

G. Pereyra, G. Tucker, J. Chorowski, L. Kaiser, and G. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *ICLR*.

V. Pham, T. Bluche, C. Kermorvant, and J. Louradour. 2014. Dropout improves recurrent neural networks for handwriting recognition. In *Frontiers in Handwriting Recognition*.

M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. 2016. Sequence level training with recurrent neural networks. In *ICLR*.

S. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. 2017. Self-critical sequence training for image captioning. In *CVPR*.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*.

I. Sutskever, O. Vinyals, and Q. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. 2016. Rethinking the inception architecture for computer vision. In *CVPR*.

R. Vedantam, C. Zitnick, and D. Parikh. 2015. CIDEr: Consensus-based image description evaluation. In *CVPR*.

O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. 2015. Show and tell: A neural image caption generator. In *CVPR*.

Z. Xie, S. Wang, J. Li, D. Lévy, A. Nie, D. Jurafsky, and A. Ng. 2017. Data noising as smoothing in neural network language models. In *ICLR*.

K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*.

Z. Yang, Y. Yuan, Y. Wu, R. Salakhutdinov, and W. Cohen. 2016. Encode, review, and decode: Reviewer module for caption generation. In *NIPS*.

T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei. 2017. Boosting image captioning with attributes. In *ICLR*.

Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. 2016. Image captioning with semantic attention. In *CVPR*.