Assessing Non-autoregressive Alignment in Neural Machine Translation via Word Reordering

Tse Chun Hin
Dept. of TIIS
Hong Kong Baptist University
cschtse
@comp.hkbu.edu.hk

Ester S.M. Leung
Asia Institute
The University of Melbourne
esther.leung
@unimelb.edu.au

William K. Cheung
Dept. of Computer Science
Hong Kong Baptist University
william
@comp.hkbu.edu.hk

Abstract

Recent work on non-autoregressive neural machine translation (NAT) that leverages alignment information to explicitly reduce the modality of target distribution has reported comparable performance with counterparts that tackle multi-modality problem by implicitly modeling dependencies. Effectiveness in handling alignment is vital for models that follow this approach, where a token reordering mechanism is typically involved and plays a vital role. We review the reordering capability of the respective mechanisms in recent NAT models, and our experimental results show that their performance is sub-optimal. We propose to learn a non-autoregressive language model (NALM) based on transformer which can be combined with Viterbi decoding to achieve better reordering performance. We evaluate the proposed NALM using the PTB dataset where sentences with words permuted in different ways are expected to have their ordering recovered. Our empirical results show that the proposed method can outperform the state-of-the-art reordering mechanisms under different word permutation settings, with a 2-27 BLEU improvement, suggesting high potential for word alignment in NAT.

1 Introduction

Non-autoregressive neural machine translation (NAT) (Gu et al., 2018) takes advantage of the parallel architecture of transformer (Vaswani et al., 2017) to alleviate the translation latency issue in neural machine translation (NMT), achieving significant speed-up. Yet it suffers from the multi-modality problem, where a target token could be a result of different possible translations. Word order errors are often resulted as compared to the autoregressive counterparts (Du et al., 2021), arising from the lack of dependency amongst target tokens in NAT models.

Some recently proposed NAT models can achieve comparable performance to autoregressive models. This can be attributed to various approaches that reduce the dependency in handling word order errors via word alignment mechanisms (Gu and Kong, 2021). In particular, latent variables and alignments have been adopted for implicitly modelling the dependencies among the target tokens (Song et al., 2021). While the latent alignment approach assumes monotonic alignment between the source and target language pair when handling token shifts in the output space (Gu and Kong, 2021), explicit modality reduction methods (Zhou et al., 2020; Shu et al., 2020; Ran et al., 2021; Song et al., 2021) on the other hand sought to directly align the source and target language pair. Despite some previous work being sub-optimal, recent work in this direction achieves state-of-the-art (sota) results rivaling that of implicit dependency modeling methods.

Establishing explicit alignment between tokens in parallel sentences of source and target languages typically involves fertility prediction and token reordering prediction. In this paper, we focus on the latter and argue that improving the reordering performance can contribute greatly towards the performance of NAT models. With the sole exception of Shu et al., 2020, architectural design of the aforementioned NAT models includes a reordering sub-module as a key component. We therefore set forward to review in detail the capabilities of the various reordering mechanisms proposed in the NAT models. We then propose a novel way to achieve the reordering prediction by learning a non-autoregressive language model (NALM) based on transformer with Viterbi decoding (Viterbi, 1967) combined.

We evaluated the reordering sub-modules extracted from the various NAT models and variants of our proposed NALM using the PTB dataset (Marcus et al., 1993) where sentences with words permuted in different ways are expected to have their ordering recovered. In particular, we adopt

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 2327–2333
December 7-11, 2022 ©2022 Association for Computational Linguistics

different degrees of permutation to mimic various levels of monotonicity (or reordering difficulty) between the source and target sentences. Our experimental results show that the proposed NALM achieves significant and consistent improvement compared to the reordering sub-modules extracted from explicit modality reductionist NAT models in all word permutation settings. Our experiment also advances the sota performance of the word reordering task in low beam setting and achieves comparable performance with autoregressive models even in high beam setting (b=64) while maintaining a constant time complexity.

2 Non-autoregressive Language Modelling

In this section, we will first provide the formulation of the word reordering task and then present our proposed solution by taking a non-autoregressive language modelling approach.

2.1 Problem definition

The word reordering problem is formulated as:

$$P(Y|Y') = P(y_0, y_1, ..., y_T | y_{\pi(0)}, y_{\pi(1)}, ..., y_{\pi(T)}) \tag{1}$$

where $Y' = y_{\pi(0)}, y_{\pi(1)}, ..., y_{\pi(T)}$ is a permutation of Y. We first follow the previous word reordering work (Hasler et al., 2017), in which we remove the permutation information and learn to recover the order of sequence Y from the corresponding bag of words $\{Y\}$. The formulation is thus revised as:

$$P(Y|\{Y\}) = P(y_0, y_1, ..., y_T | \{y_0, y_1, ..., y_T\}) \tag{2}$$

where $\{y_0, y_1, ..., y_T\}$ denotes a set of Y. This can be approximated as:

$$P(Y|\{Y\}) = \prod_{t=1}^{T} P(y_t | y_{t-1}, \{Y\}) \tag{3}$$

so that each token's probability is now conditioned to the token immediately preceding it as well as to the entire bag of tokens in the sequence.

2.2 NALM

The training setup of a standard transformer decoder in NMT naturally conforms to the above formulation, as it learns the conditional probability $P(y_t|y_{t-1}, X)$. Since our model does not involve translating from X to Y, the inter-attention layer can thus be removed and the decoder becomes a standard transformer encoder. However we still need to include bag of words $\{Y\}$ into our modeling. This can be achieved by replacing the causal attention with full attention and removing the position embedding of the input in the model. We further replace the output layer with a pointer network to constraint the output space to only the tokens (including repetition) within the concerned sequence. The entire model is thus formulated as:

$$\begin{aligned} H &= transformer(bos \oplus Y) \\ O &= H \cdot (Y \oplus eos) \end{aligned} \tag{4}$$

where bos and eos refer to the beginning and the end of sentence tokens respectively. O is the output of the pointer network. We train the model by minimizing the cross entropy.

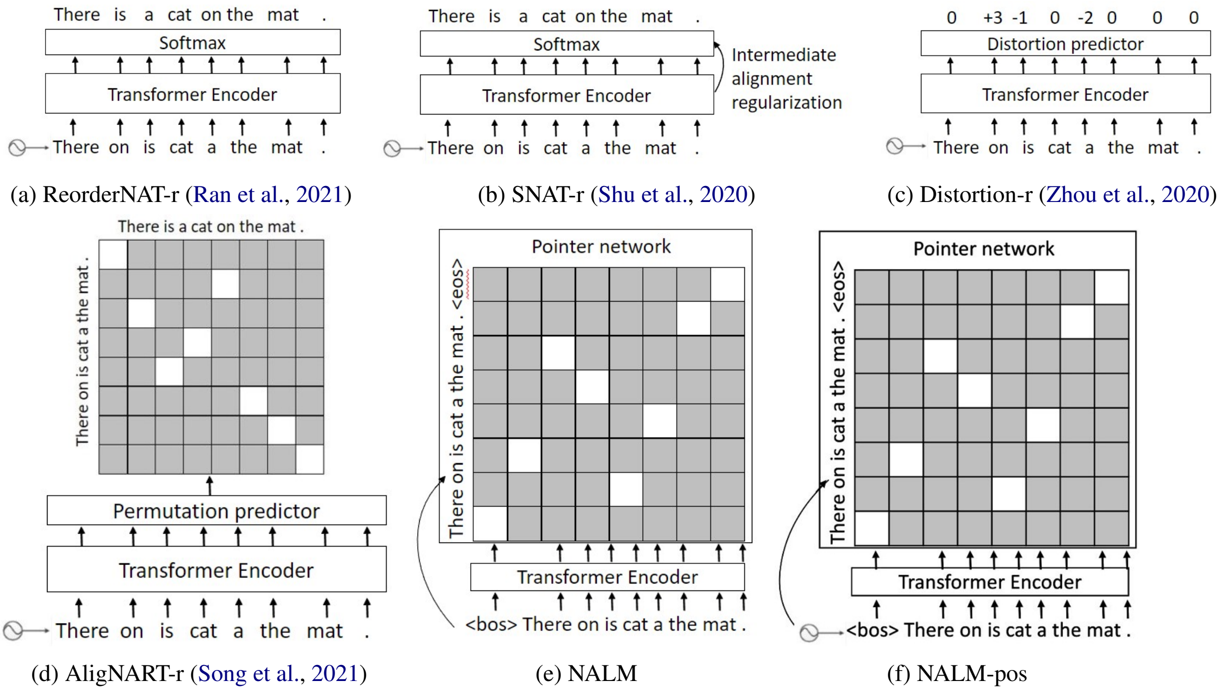
In the pointer network, we utilize the input sequence as the vocabulary and output a non-normalized matrix which can yield a probability matrix via softmax (see Figure 1(e) and 1(f)). This output probability matrix can be viewed as a trellis containing the transition probabilities of each input token to the rest of its neighbours. The optimal path that traverses this trellis would guarantee the most probable sequence of transitions using the well-known Viterbi algorithm (Viterbi, 1967).

2.3 NALM-pos

The NALM learns a probability distribution of sequences which essentially allows its reconstruction by considering the input as a bag of tokens. Yet, modeling the underlying permutation mechanism between sequences and their permutations could also be useful for achieving better reordering. In order to capture the mechanism as well, we extend NALM with position information. Our extended model NALM-pos learns the following probabilities:

$$P(Y|Y') = \prod_{t=1}^{T} P(y_t | y'_t, Y') \tag{5}$$

The advantage of $P(Y|Y')$ over $P(Y|\{Y\})$ is that it retains certain ordering information of the sequence albeit permutation from the ground truth order. When much of the permutation order resembles the ground truth order, retaining such position information will significantly reduces the complexity of the learning task. In NAT, the reordering sub-module receives a transformed source sentence as input, which generally still follows source word order. The target of the sub-module on the other


There is a cat on the mat .
Softmax
Transformer Encoder
There on is cat a the mat .
Intermediate alignment regularization
0 +3 -1 0 -2 0 0 0
Distortion predictor
Permutation predictor
Pointer network
There on is cat a the mat . <eos>
<bos> There on is cat a the mat .

(a) ReorderNAT-r (Ran et al., 2021)

(b) SNAT-r (Shu et al., 2020)

(c) Distortion-r (Zhou et al., 2020)

(d) AligNART-r (Song et al., 2021)

(e) NALM

(f) NALM-pos

Figure 1: Architectures of the reordering components adopted in the existing works and our proposed models. For example, ReorderNAT-r denotes the reordering mechanism in ReorderNAT.

hand follows the target word order. Yet, these word orders are often shared between languages. The more similar these orders are, the more monotonic the two languages are. Even in less monotonic language pair such as JA-EN, orders are shared to some extent. This further accounts for the importance of incorporating position information.

3 Experimental Setup

We use the English Penn Treebank data (Marcus et al., 1993) in our evaluation, preprocessed (in various ways) as described in the section.

3.1 Data and evaluation

Following Hasler et al., 2017, we conduct experiments on the data preprocessed as in Schmaltz et al., 2016 for fair comparison.[1] This dataset is fully shuffled on the token level, and we refer it as ptb2016. We further create 3 datasets based on the preprocessed data to simulate reordering data that would more likely be encountered in NAT alignment. They simulate reordering data of varying difficulty. We start by ngramizing the sentences to simulate phrases commonly found in phrase-based statistical machine translation. We argue that the different ordering between parallel sentences of two languages involves predominantly movement of these phrases (local orderings), and therefore permuting these ngrams will provide datasets which resemble better the challenges faced in real alignment during NAT. We employ two methods in permuting these ngrams, either by randomly permuting a percentage of them (0.4 and 0.6), or by adjacently displace-and-combine pairs of ngrams recurrently based on a preset probability (0.5). We refer them as the r04, r06 and d05 datasets respectively. We use quadgram with backoff during ngramization and ngrams with count above 2.

3.2 Model settings

For all the models, we follow the transformer_base_v3 hparams set as defined in tensor2tensor (Vaswani et al., 2018). We train the model with a total of 100k steps with a batch size of $65,536$. Evaluation is done via the t2t-bleu script and we report case sensitive BLEU scores as well as METEOR scores. Following Schmaltz et al., 2016, we use a vocabulary of $16,161$ including two different unk tokens.

We report model parameter size in Table 1 for reference and include short and long samples of reordering results in Tables 4 and 5 for comparison. We use 2 GTX 1080 ti for model training in all our experiments and the multistep function provided in tensor2tensor was

[1] We thank the authors for help to reproduce their results.

used to overcome the memory problem posted in training. Our source code is available at https://github.com/colmantse/NALM.git

Model	No. of parameters
ReorderNAT-r	27,177,472
SNAT-r	31,902,208
AligNART-r	37,418,496
Distortion-r	28,354,688
NALM	27,177,472
NALM-pos	27,177,472

Table 1: Models' parameter size.

3.3 Benchmark models

We describe the reordering modules extracted from the existing NAT models involved in the evaluation as follow. We apply the Hungarian algorithm (Kuhn, 1955) to the output matrix for all the reordering modules to obtain the final order of the permuted sequence.

ReorderNAT-r

The reordering module of ReorderNAT (Ran et al., 2021) is a transformer decoder which we replace with an encoder given the monolingual setting of the word reordering task, otherwise unchanged.

SNAT-r

The SNAT (Shu et al., 2020) is an explicit modality reductionist model. Its alignment mechanism is achieved via latent regularization to the model's transformer decoder. We replace the decoder with an encoder as suitable for word reordering task and upon it implement the regularization.

Distortion-r

The distortion model (Zhou et al., 2020) makes use of a distortion predictor to predict alignment by taking encoder output as input. We retain their encoder and distortion predictor for the word reordering task, with the fertility predictor and the decoder removed. In their work, they tried both absolute position and relative position information in their distortion predictor. We only experiment with the relative position distortion predictor because of its superior performance as reported.

AligNART-r

AligNART (Song et al., 2021) is currently the sota method using the explicit NAT approach. Its performance is also comparable to the sota in the field of NAT. For adaptation to the reordering task, we remove the decoder as well as the duplication predictor and the grouping predictor in the aligner, leaving only the permutation predictor. We use a 6-layer encoder to fit the task setting. We train the adapted model only by minimizing the KL-divergence, i.e. the permutation predictor loss in the original work.[2]

Model	**BLEU**	**Model**	**BLEU**
AR (beam=5)		*NAR*	
n-gram*	23.3	ReorderNAT-r	15.21
RNNLM*	24.5	SNAT-r	17.03
bag2seq*	33.4	AligNART-r	7.54
AttM*	<u>34.89</u>	Distortion-r	7.47
Transformer	34.14	NALM	**35.86**
		NALM-pos	31.16

Table 2: BLEU scores for the word-ordering task on the ptb2016 dataset. Other than AttM, whose performance is reported from Tao et al., 2021, all previous works (indicated by *) are reported from Hasler et al., 2017. Autoregressive (AR) models are listed on the left while non-autoregressive (NAR) models are listed on the right.

4 Results

4.1 Word reordering on the Penn Treebank

Table 2 shows that NALM can outperform all other reordering mechanisms adapted from the existing NAT models by at least 8 BLEU. It furthermore surpasses the transformer baseline (b=5) by 1 BLEU. Since this paper aims to study reordering mechanism in NAT, we do not include baseline transformer's performance in higher beam settings, as the lengthy decoding time would defeat the purpose of fast and efficient NAT approaches. However even when pitched against past works with higher beam settings (e.g., b=64), NALM still compares. [3] Amongst the benchmark models, Song et al., 2021 fails to converge during training while Zhou et al., 2020 fails to recover any meaningful ordering even when fully trained. The disappointing performance of the adapted reordering mechanisms can be attributed to their deficiency in recovering sequences from random ordering, and suggesting a heavy reliance on shared local orderings between the input and output sequences. Notably, the performance of NALM-pos is not as good as that of NALM. This illustrates that reordering models clearly expect

[2] We thank the authors for help on the reimplementation.

[3] bag2seq (b=64) was 36.2, 0.34 BLEU higher than ours.

the aforementioned shared local orderings. When the said ordering information is removed, positional information, considered by NALM-pos and all adapted models, would only confuse learning and hamper performance.

4.2 On different degrees of permutation

The previous experiment assumes that input is shuffled at the token level. To evaluate on ordering tasks that better reflect real reordering situations in NAT alignment, we further conduct testing on r04, r06 and d05 datasets which permute the dataset at n-gram level. Table 3 shows that NALM-pos's performance leads all other adapted reordering mechanisms by at least 11 BLEU. According to the experimental results, the more permuted the data, the poorer the performance of all the models, except NALM, which is invariant to input permutation. Interestingly, this simple design already outperforms all other adapted mechanisms for all datasets by 2-10 BLEU, showing great versatility in all settings of word permutation. After augmenting it with positional information, NALM-pos advances performance further by 7-15 BLEU. We also report METEOR score in this experiment, and it more or less reflects the same trend as in BLEU.

We note that adapted reordering mechanism of the sota NAT model does not perform well when it stands alone, suggesting the need of further investigation. As for Zhou et al., 2020, upon closer inspection, we find that their reordering mechanism simply learns to copy. This also explains its poor performance in the first experiment, as it simply copied the random input permutation which score terribly against the ground truth sequence.

5 Conclusion

In this paper, we review reordering mechanisms of NAT models that directly model alignment using various settings of word permutation. We propose a non-autoregressive language model which outperforms in low-beam and competes with in higher-beam setting sota autoregressive models. Our extended model further achieves significant improvement over all adapted NAT reordering mechanisms in datasets of varying difficulty that reasonably resemble the reordering task encountered in NAT alignment. Performance of existing reordering mechanisms in NAT models vary according to our experiment results, implying that more effort would be required in this area.

Dataset	Model	BLEU	METEOR
r04	ReorderNAT-r	31.98	81.5
	SNAT-r	33.53	82.9
	AligNART-r	23.97	75.9
	NALM	35.86	83.6
	NALM-pos	**50.96**	**89.8**
r06	ReorderNAT-r	20.46	73.0
	SNAT-r	24.11	76.6
	AligNART-r	29.48	78.4
	NALM	35.86	**83.6**
	NALM-pos	**42.87**	86.8
d05	ReorderNAT-r	23.71	76.3
	SNAT-r	25.89	78.3
	AligNART-r	20.52	72.8
	NALM	35.86	83.6
	NALM-pos	**46.78**	**88.1**

Table 3: BLEU and METEOR scores for the word reordering task on the r04, r06, and d05 datasets. Sample results are also provided in Tables 4 and 5 in appendix A. Note that Distortion-r is removed from the table as it learns only to copy from the input permutation.

6 Limitations

We acknowledge that our experiment is a simplification to the real reordering problem in NAT alignment. Results can only partially reflect the capability of concerned models in optimal conditions. A better experiment would be to include also subword vocabularies in bilingual settings, with supervised reordering data. We leave this to our future work.

References

Cunxiao Du, Zhaopeng Tu, and Jing Jiang. 2021. Order-agnostic cross entropy for non-autoregressive machine translation. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2849–2859. PMLR.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, Canada, April 30-May 3, 2018, Conference Track Proceedings*.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 120–133, Online. Association for Computational Linguistics.

Eva Hasler, Felix Stahlberg, Marcus Tomalin, Adrià de Gispert, and Bill Byrne. 2017. A comparison of neural models for word ordering. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 208–212, Santiago de Compostela, Spain. Association for Computational Linguistics.

Harold W. Kuhn. 1955. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Qiu Ran, Yankai Lin, Peng Li, and Jie Zhou. 2021. Guiding non-autoregressive neural machine translation decoding with reordering information. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13727–13735.

Allen Schmaltz, Alexander M. Rush, and Stuart Shieber. 2016. Word ordering without syntax. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2319–2324, Austin, Texas. Association for Computational Linguistics.

Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. 2020. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8846–8853.

Jongyoon Song, Sungwon Kim, and Sungroh Yoon. 2021. AligNART: Non-autoregressive neural machine translation by jointly learning to estimate alignment and translate. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1–14, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chongyang Tao, Shen Gao, Juntao Li, Yansong Feng, Dongyan Zhao, and Rui Yan. 2021. Learning to organize a bag of words into sentences with neural networks: An empirical study. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1682–1691, Online. Association for Computational Linguistics.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

A. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269.

Long Zhou, Jiajun Zhang, Yang Zhao, and Chengqing Zong. 2020. Non-autoregressive neural machine translation with distortion model. In *Natural Language Processing and Chinese Computing*, pages 403–415, Cham. Springer International Publishing.

A Appendix

Ground Truth	Then he jumped into the market :	“ I spent $ N million in the last half-hour . ”
Input permutation	Then into the market jumped : he	“ I spent $ N million . ” in the last half-hour
ReorderNAT-r	Then the he : jumped market into	“ I spent $ half-hour N million in ” last . the
SNAT-r	Then he market jumped into : the	“ I spent $ N million . in the last half-hour ”
AligNART-r	Then into the : jumped he market	“ I spent $ N million half-hour ” in the last .
Distortion-r	Then into the market jumped : he	“ I spent $ N million . ” in the last half-hour
NALM	jumped into the market : Then he	“ I spent $ N million in the last half-hour . ”
NALM-pos	Then he jumped into the market :	“ I spent $ N million in the last half-hour . ”

Table 4: Short samples from the r06 dataset.

Ground Truth	“ The last crash taught institutional investors that they have to be long-term holders , and that they ca n’t react to short-term events , good or bad , ” said Stephen L. UNK , senior vice president for the pension consultants Wilshire Associates in Santa Monica , Calif .
Input permutation	that “ The last crash , ” said Stephen that L. have to be long-term holders , and pension they ca n’t react to short-term events , good or Wilshire Associates in taught UNK , senior vice president for the they consultants bad institutional investors Santa Monica , Calif .
ReorderNAT-r	“ The last , Santa crash , Stephen that said L. react that to have long-term holders , pension they and they ca n’t short-term to be events or good Wilshire Associates in , taught bad senior vice president UNK consultants institutional investors for the Monica , ” Calif .
SNAT-r	“ The last , L. said bad investors that Stephen that have be react to long-term crash , holders and they ca n’t short-term , pension they events good or Wilshire Associates in UNK , senior taught vice president for the consultants , ” to institutional Santa Monica Calif .
AligNART-r	that that “ The last , crash , said pension Stephen L. to to the Associates be consultants long-term n’t react ca short-term have events , investors or Wilshire good holders taught UNK in , , senior ” president they they for bad institutional and Santa Monica vice Calif .
Distortion-r	that “ The last crash , ” said Stephen that L. have to be long-term holders , and pension they ca n’t react to short-term events , good or Wilshire Associates in taught UNK , senior vice president for the they consultants bad institutional investors Santa Monica , Calif .
NALM	events for that they ca n’t react to the long-term and that they have good institutional investors last , senior vice president , Calif . ” said Stephen L. UNK , in short-term holders , or bad Santa Monica , “ The consultants Associates taught to be pension crash Wilshire
NALM-pos	have to be long-term holders , and pension that “ The last crash , ” said Stephen L. UNK , senior vice president for the consultants ca n’t react to short-term events , or bad institutional investors , Calif . Associates in Santa Monica Wilshire that they taught they good

Table 5: Long samples from the r04 dataset.