Is neural machine translation ready for deployment?

A case study on 30 translation directions

Marcin Junczys-Dowmunt[1,2], Tomasz Dwojak[1,2], Hieu Hoang[3]

[1]Adam Mickiewicz University in Poznań
[2]University of Edinburgh
[3]Moses Machine Translation CIC

{junczys,t.dwojak}@amu.edu.pl hieu@hoang.co.uk

Introduction

- Neural Machine Translation has been widely evaluated for known shared tasks e.g. WMT, NIST, and right now IWSLT.
- Google, SYSTRAN, Microsoft announced switch to NMT for general-domain engines.
- Translation (automation) industry news sites are buzzing with articles about NMT
- But AMTA 2016: no NMT paper in the user or government sections.
- What about in-domain, in-house training data of multi-national organizations or LSPs?

Introduction

- Neural Machine Translation has been widely evaluated for known shared tasks e.g. WMT, NIST, and right now IWSLT.
- Google, SYSTRAN, Microsoft announced switch to NMT for general-domain engines.
- Translation (automation) industry news sites are buzzing with articles about NMT
- But AMTA 2016: no NMT paper in the user or government sections.
- What about in-domain, in-house training data of multi-national organizations or LSPs?

Introduction

- Neural Machine Translation has been widely evaluated for known shared tasks e.g. WMT, NIST, and right now IWSLT.
- Google, SYSTRAN, Microsoft announced switch to NMT for general-domain engines.
- Translation (automation) industry news sites are buzzing with articles about NMT
- But AMTA 2016: no NMT paper in the user or government sections.
- What about in-domain, in-house training data of multi-national organizations or LSPs?

Introduction

- Neural Machine Translation has been widely evaluated for known shared tasks e.g. WMT, NIST, and right now IWSLT.
- Google, SYSTRAN, Microsoft announced switch to NMT for general-domain engines.
- Translation (automation) industry news sites are buzzing with articles about NMT
- But AMTA 2016: no NMT paper in the user or government sections.
- What about in-domain, in-house training data of multi-national organizations or LSPs?

Introduction

- Neural Machine Translation has been widely evaluated for known shared tasks e.g. WMT, NIST, and right now IWSLT.
- Google, SYSTRAN, Microsoft announced switch to NMT for general-domain engines.
- Translation (automation) industry news sites are buzzing with articles about NMT
- But AMTA 2016: no NMT paper in the user or government sections.
- What about in-domain, in-house training data of multi-national organizations or LSPs?

Introduction

- Neural Machine Translation has been widely evaluated for known shared tasks e.g. WMT, NIST, and right now IWSLT.
- Google, SYSTRAN, Microsoft announced switch to NMT for general-domain engines.
- Translation (automation) industry news sites are buzzing with articles about NMT
- But AMTA 2016: no NMT paper in the user or government sections.
- What about in-domain, in-house training data of multi-national organizations or LSPs?

Is neural machine translation ready for deployment?

1. Is quality a problem?

2. Is translation speed a problem?

The United Nations Parallel Corpus v1.0

- United Nations Parallel Corpus v1.0 [Ziemski et al., 2016] as an example for in-domain training data used for in-house translation service;
- Human translated UN documents from 1990 to 2014;
- Arabic, Chinese, English, French, Russian, and Spanish;
- Fully aligned subcorpus across all UN languages available, 30 translation directions in total;
- Documents released in 2015 distributed as dev and test sets.

Documents	Lines	English Tokens
86,307	11,365,709	334,953,817

The United Nations Parallel Corpus v1.0

- United Nations Parallel Corpus v1.0 [Ziemski et al., 2016] as an example for in-domain training data used for in-house translation service;
- Human translated UN documents from 1990 to 2014;
- Arabic, Chinese, English, French, Russian, and Spanish;
- Fully aligned subcorpus across all UN languages available, 30 translation directions in total;
- Documents released in 2015 distributed as dev and test sets.

Documents	Lines	English Tokens
86,307	11,365,709	334,953,817

The United Nations Parallel Corpus v1.0

- United Nations Parallel Corpus v1.0 [Ziemski et al., 2016] as an example for in-domain training data used for in-house translation service;
- Human translated UN documents from 1990 to 2014;
- Arabic, Chinese, English, French, Russian, and Spanish;
- Fully aligned subcorpus across all UN languages available, 30 translation directions in total;
- Documents released in 2015 distributed as dev and test sets.

Documents	Lines	English Tokens
86,307	11,365,709	334,953,817

The United Nations Parallel Corpus v1.0

- United Nations Parallel Corpus v1.0 [Ziemski et al., 2016] as an example for in-domain training data used for in-house translation service;
- Human translated UN documents from 1990 to 2014;
- Arabic, Chinese, English, French, Russian, and Spanish;
- Fully aligned subcorpus across all UN languages available, 30 translation directions in total;
- Documents released in 2015 distributed as dev and test sets.

Documents	Lines	English Tokens
86,307	11,365,709	334,953,817

The United Nations Parallel Corpus v1.0

- United Nations Parallel Corpus v1.0 [Ziemski et al., 2016] as an example for in-domain training data used for in-house translation service;
- Human translated UN documents from 1990 to 2014;
- Arabic, Chinese, English, French, Russian, and Spanish;
- Fully aligned subcorpus across all UN languages available, 30 translation directions in total;
- Documents released in 2015 distributed as dev and test sets.

Documents	Lines	English Tokens
86,307	11,365,709	334,953,817

Preprocessing

- Sentences longer than 100 words were discarded;
- Lowercased data as done in [Ziemski et al., 2016];
- Tokenized with the Moses tokenizer;
- Jieba[1] for Chinese segmentation;
- BPE subword units for NMT models (30,000).

[1] https://github.com/fxsjy/jieba

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Phrase-based Baselines

[Ziemski et al., 2016] provided BLEU scores for Moses configurations trained on the 6-way subcorpus.

- Word alignment with MIGZA++, 5x Model1, 5x HMM;
- 5-gram language model from the target parallel data;
- Default lexical reordering model;
- 5-gram operation sequence model;
- 9-gram word-class language model with word-classes produced by word2vec [Mikolov et al., 2013];
- Significance pruned [Johnson et al., 2007] phrase-table;
- For decoding, cube-pruning algorithm with stack size and cube-pruning pop limits of 1,000.

Neural translation systems

- Attentional encoder-decoder [Bahdanau et al., 2014], implemented in Nematus [Sennrich et al., 2016];
- Mini-batches of size 40, maximum sentence length of 100, word embeddings of size 500, and hidden layers of size 1024.
- We clip the gradient norm to 1.0 [Pascanu et al., 2013].
- Models were trained with Adadelta [Zeiler, 2012], reshuffling the training corpus between epochs.
- Trained for 1.2M iterations, checkpoint every 30K iterations;
- Last four model checkpoints for ensembling;
- Trained for 4 epochs/8 days on a Nvidia GTX 1080;
- Total time: 320 days, up to 16 GPUs (ca. 2.5 months).

Neural translation systems

- Attentional encoder-decoder [Bahdanau et al., 2014], implemented in Nematus [Sennrich et al., 2016];
- Mini-batches of size 40, maximum sentence length of 100, word embeddings of size 500, and hidden layers of size 1024.
- We clip the gradient norm to 1.0 [Pascanu et al., 2013].
- Models were trained with Adadelta [Zeiler, 2012], reshuffling the training corpus between epochs.
- Trained for 1.2M iterations, checkpoint every 30K iterations;
- Last four model checkpoints for ensembling;
- Trained for 4 epochs/8 days on a Nvidia GTX 1080;
- Total time: 320 days, up to 16 GPUs (ca. 2.5 months).

Neural translation systems

- Attentional encoder-decoder [Bahdanau et al., 2014], implemented in Nematus [Sennrich et al., 2016];
- Mini-batches of size 40, maximum sentence length of 100, word embeddings of size 500, and hidden layers of size 1024.
- We clip the gradient norm to 1.0 [Pascanu et al., 2013].
- Models were trained with Adadelta [Zeiler, 2012], reshuffling the training corpus between epochs.
- Trained for 1.2M iterations, checkpoint every 30K iterations;
- Last four model checkpoints for ensembling;
- Trained for 4 epochs/8 days on a Nvidia GTX 1080;
- Total time: 320 days, up to 16 GPUs (ca. 2.5 months).

Neural translation systems

- Attentional encoder-decoder [Bahdanau et al., 2014], implemented in Nematus [Sennrich et al., 2016];
- Mini-batches of size 40, maximum sentence length of 100, word embeddings of size 500, and hidden layers of size 1024.
- We clip the gradient norm to 1.0 [Pascanu et al., 2013].
- Models were trained with Adadelta [Zeiler, 2012], reshuffling the training corpus between epochs.
- Trained for 1.2M iterations, checkpoint every 30K iterations;
- Last four model checkpoints for ensembling;
- Trained for 4 epochs/8 days on a Nvidia GTX 1080;
- Total time: 320 days, up to 16 GPUs (ca. 2.5 months).

Neural translation systems

- Attentional encoder-decoder [Bahdanau et al., 2014], implemented in Nematus [Sennrich et al., 2016];
- Mini-batches of size 40, maximum sentence length of 100, word embeddings of size 500, and hidden layers of size 1024.
- We clip the gradient norm to 1.0 [Pascanu et al., 2013].
- Models were trained with Adadelta [Zeiler, 2012], reshuffling the training corpus between epochs.
- Trained for 1.2M iterations, checkpoint every 30K iterations;
- Last four model checkpoints for ensembling;
- Trained for 4 epochs/8 days on a Nvidia GTX 1080;
- Total time: 320 days, up to 16 GPUs (ca. 2.5 months).

Neural translation systems

- Attentional encoder-decoder [Bahdanau et al., 2014], implemented in Nematus [Sennrich et al., 2016];
- Mini-batches of size 40, maximum sentence length of 100, word embeddings of size 500, and hidden layers of size 1024.
- We clip the gradient norm to 1.0 [Pascanu et al., 2013].
- Models were trained with Adadelta [Zeiler, 2012], reshuffling the training corpus between epochs.
- Trained for 1.2M iterations, checkpoint every 30K iterations;
- Last four model checkpoints for ensembling;
- Trained for 4 epochs/8 days on a Nvidia GTX 1080;
- Total time: 320 days, up to 16 GPUs (ca. 2.5 months).

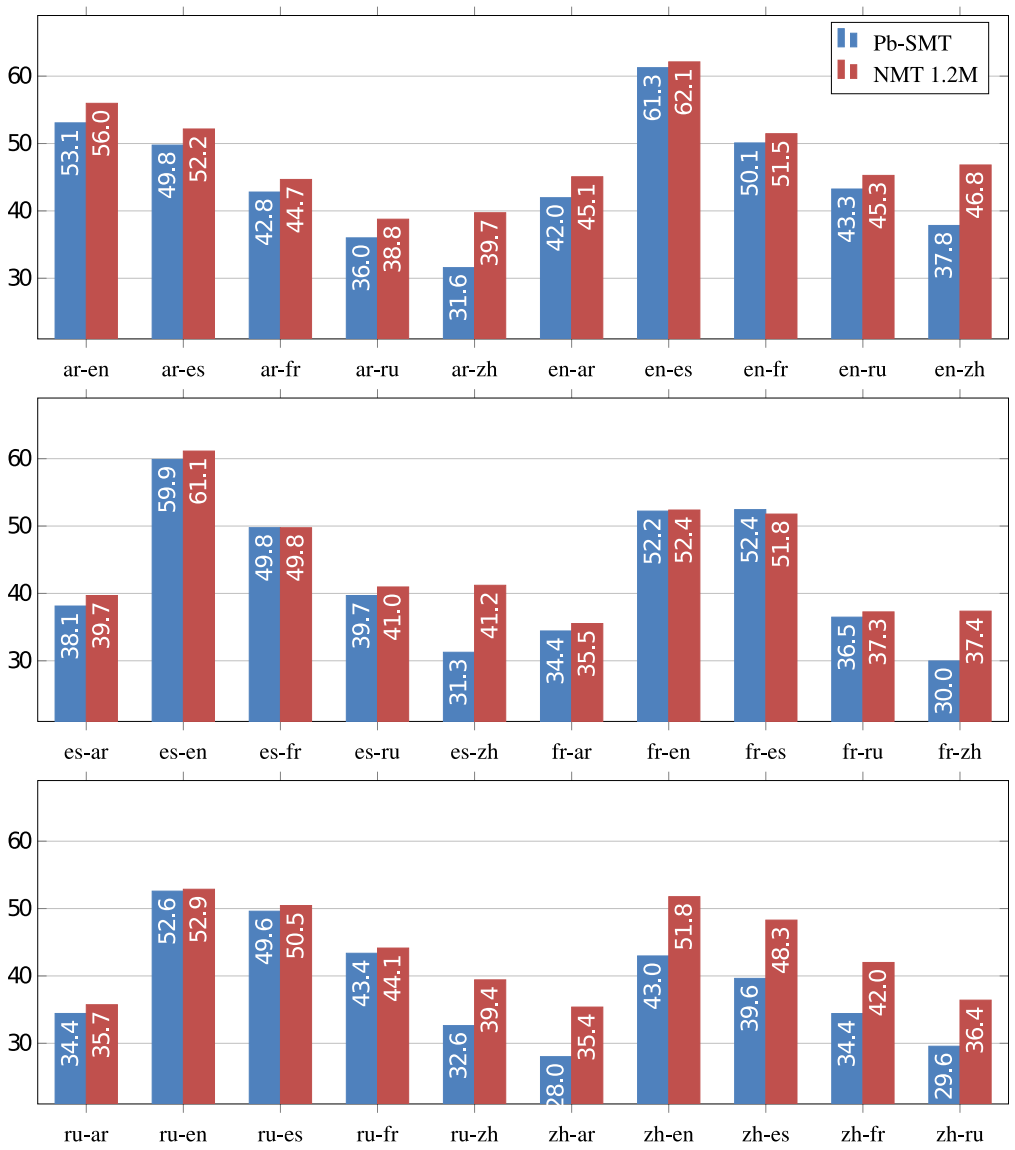
Neural translation systems

- Attentional encoder-decoder [Bahdanau et al., 2014], implemented in Nematus [Sennrich et al., 2016];
- Mini-batches of size 40, maximum sentence length of 100, word embeddings of size 500, and hidden layers of size 1024.
- We clip the gradient norm to 1.0 [Pascanu et al., 2013].
- Models were trained with Adadelta [Zeiler, 2012], reshuffling the training corpus between epochs.
- Trained for 1.2M iterations, checkpoint every 30K iterations;
- Last four model checkpoints for ensembling;
- Trained for 4 epochs/8 days on a Nvidia GTX 1080;
- Total time: 320 days, up to 16 GPUs (ca. 2.5 months).

Neural translation systems

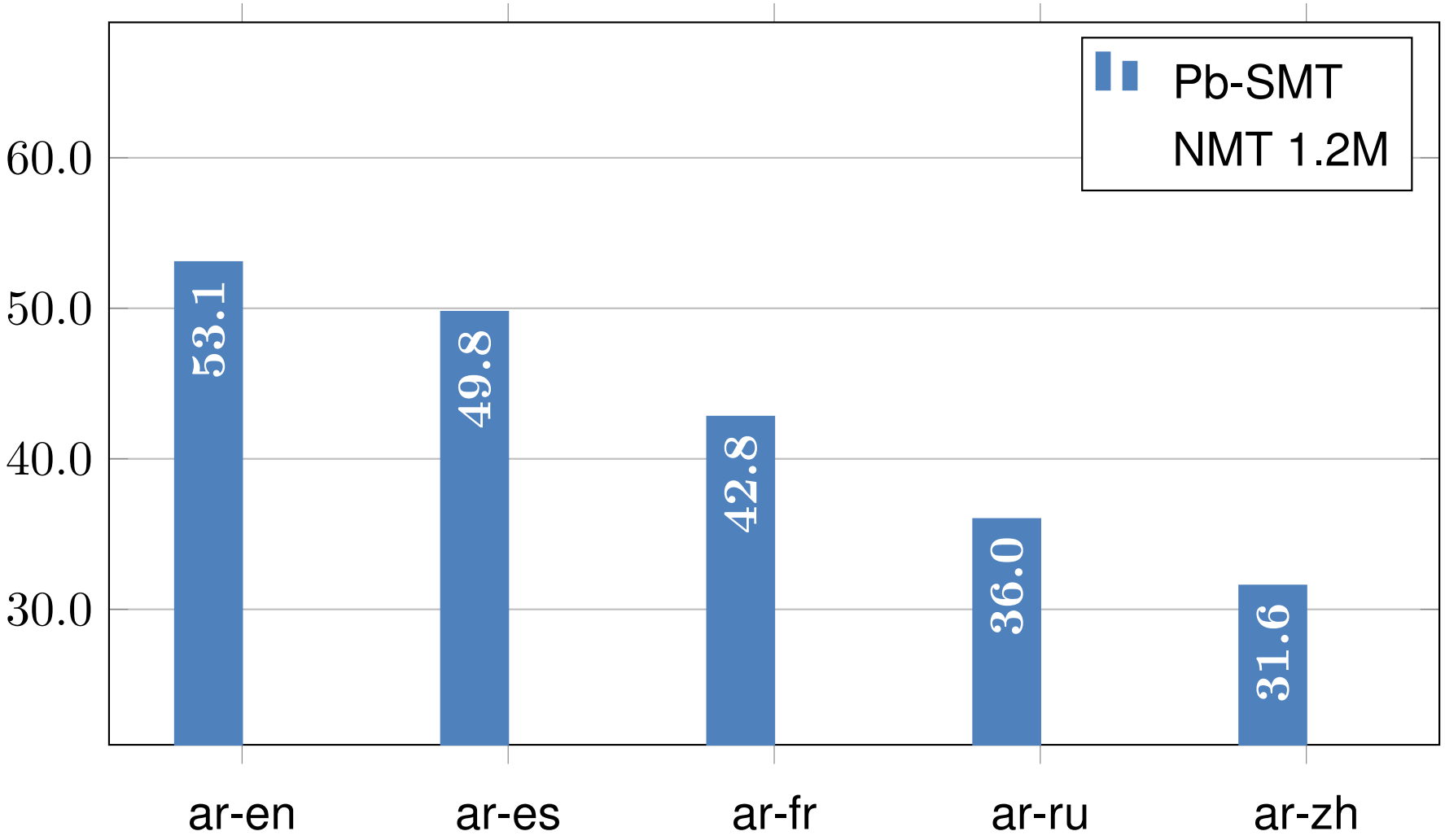
- Attentional encoder-decoder [Bahdanau et al., 2014], implemented in Nematus [Sennrich et al., 2016];
- Mini-batches of size 40, maximum sentence length of 100, word embeddings of size 500, and hidden layers of size 1024.
- We clip the gradient norm to 1.0 [Pascanu et al., 2013].
- Models were trained with Adadelta [Zeiler, 2012], reshuffling the training corpus between epochs.
- Trained for 1.2M iterations, checkpoint every 30K iterations;
- Last four model checkpoints for ensembling;
- Trained for 4 epochs/8 days on a Nvidia GTX 1080;
- Total time: 320 days, up to 16 GPUs (ca. 2.5 months).

PB-SMT vs. NMT


Pb-SMT
NMT 1.2M
60
50
40
30
53.1 56.0
49.8 52.2
42.8 44.7
36.0 38.8
31.6 39.7
42.0 45.1
61.3 62.1
50.1 51.5
43.3 45.3
37.8 46.8
ar-en ar-es ar-fr ar-ru ar-zh en-ar en-es en-fr en-ru en-zh
38.1 39.7
59.9 61.1
49.8 49.8
39.7 41.0
31.3 41.2
34.4 35.5
52.2 52.4
52.4 51.8
36.5 37.3
30.0 37.4
es-ar es-en es-fr es-ru es-zh fr-ar fr-en fr-es fr-ru fr-zh
34.4 35.7
52.6 52.9
49.6 50.5
43.4 44.1
32.6 39.4
28.0 35.4
43.0 51.8
39.6 48.3
34.4 42.0
29.6 36.4
ru-ar ru-en ru-es ru-fr ru-zh zh-ar zh-en zh-es zh-fr zh-ru

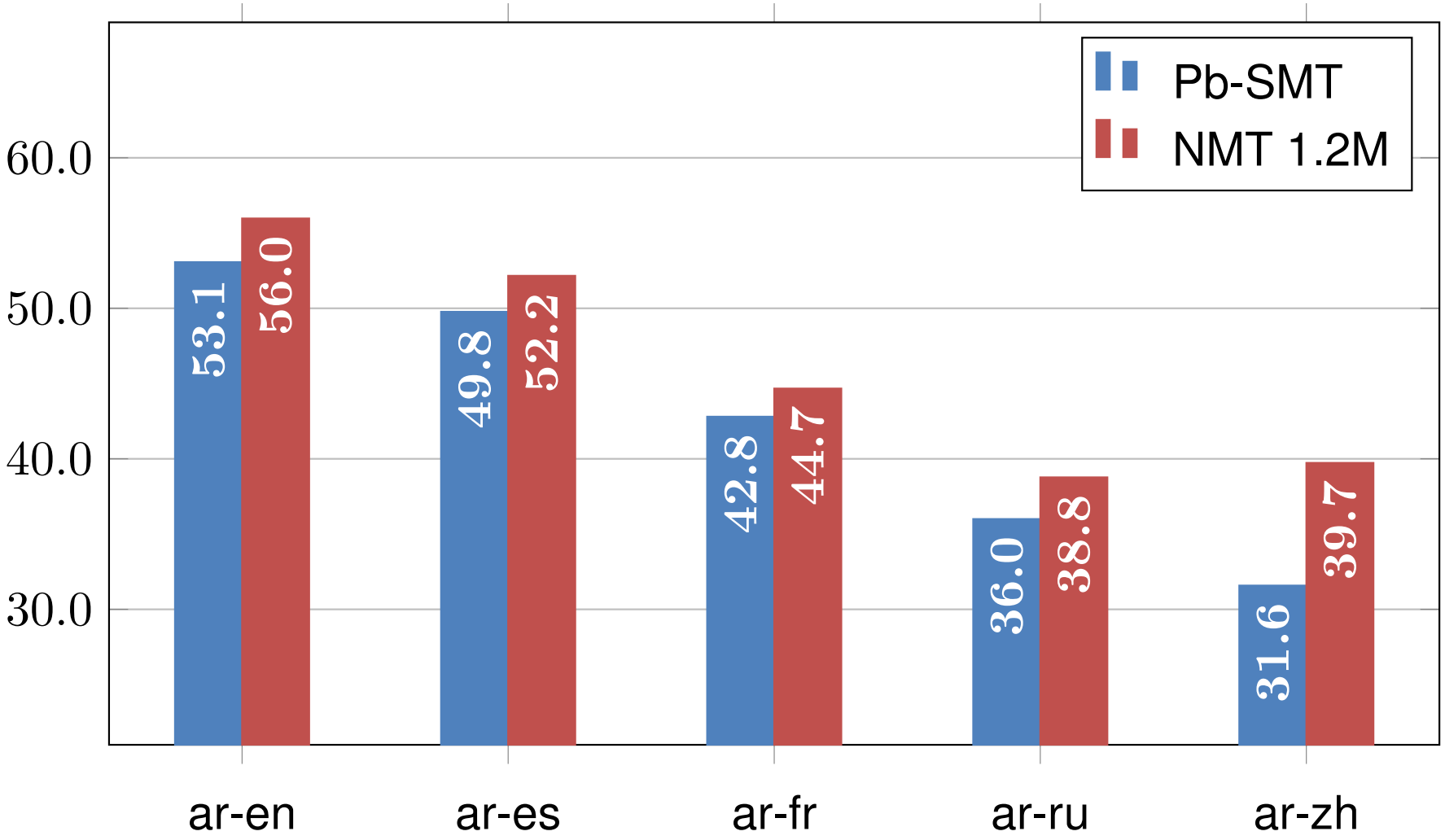
PB-SMT vs. NMT

Out of Arabic



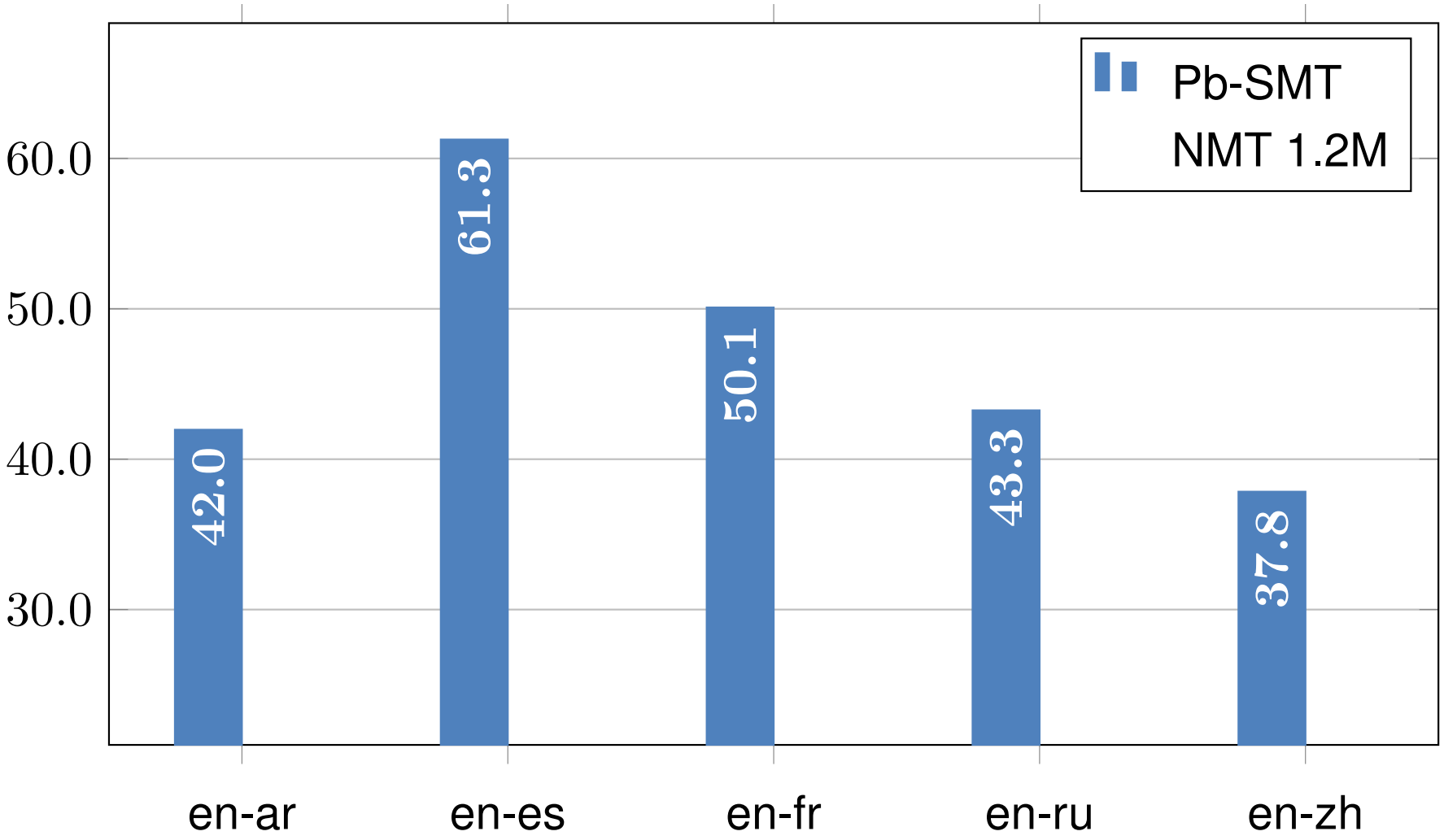
PB-SMT vs. NMT

Out of Arabic



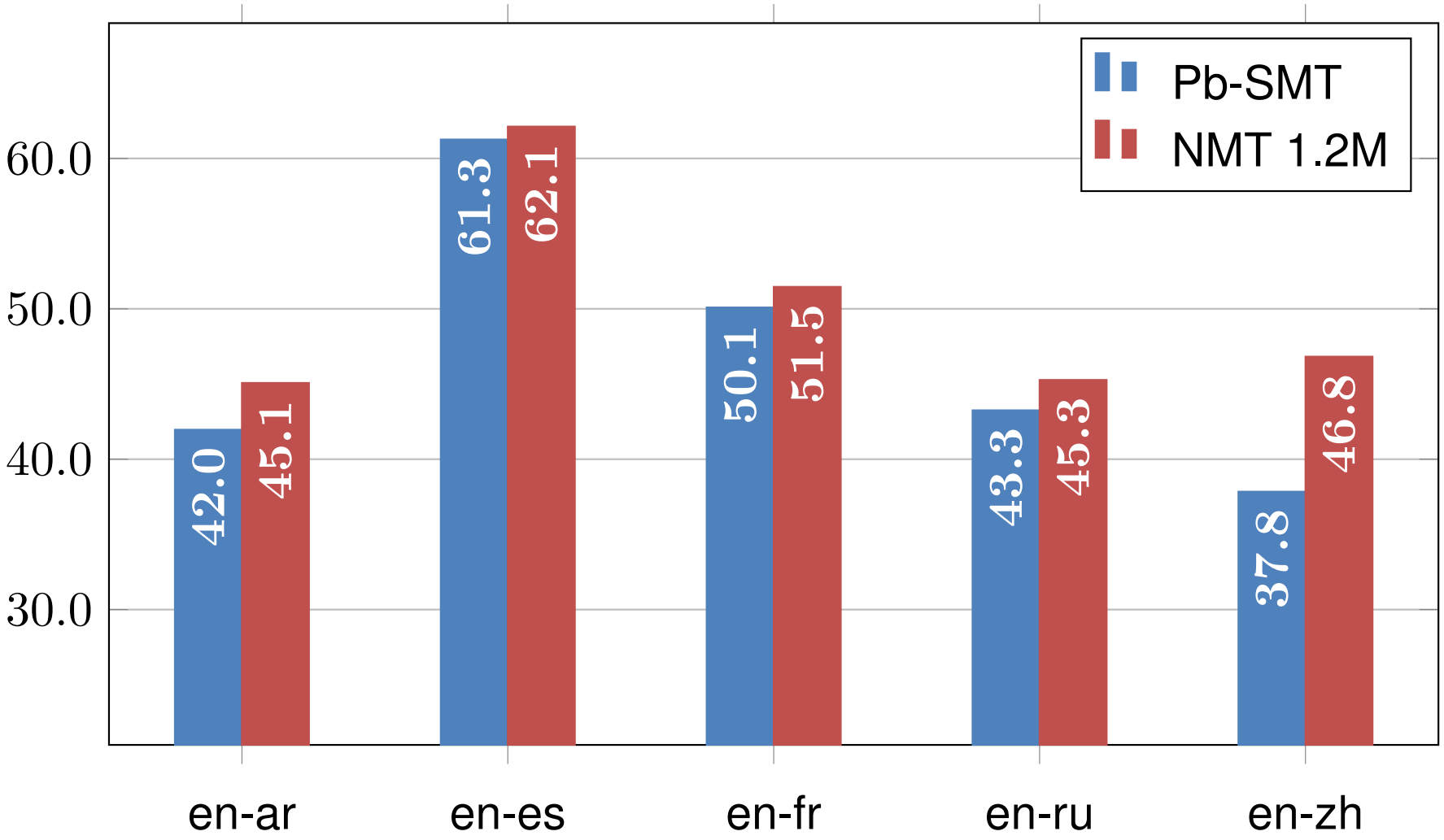
PB-SMT vs. NMT

Out of English



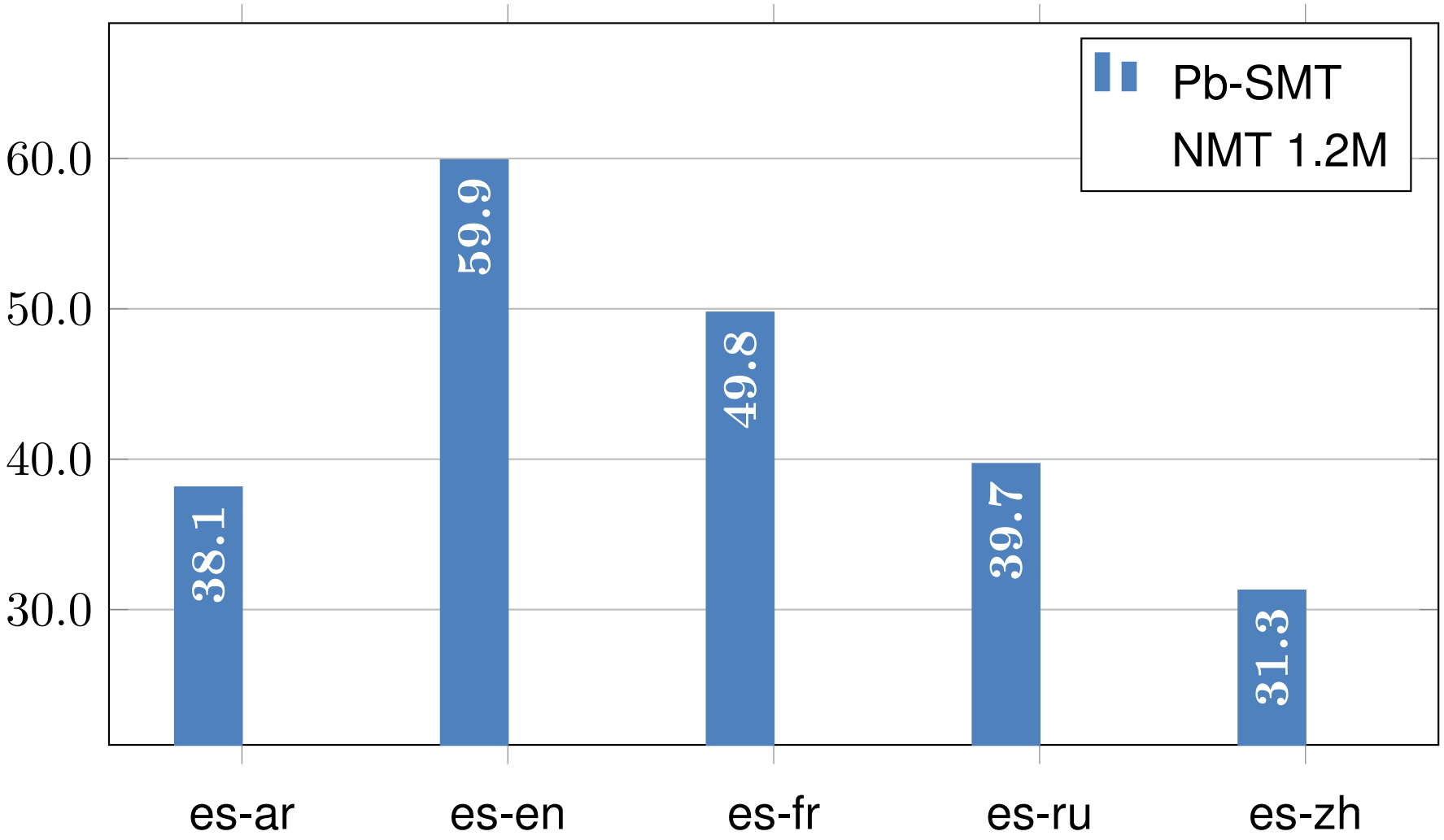
PB-SMT vs. NMT

Out of English



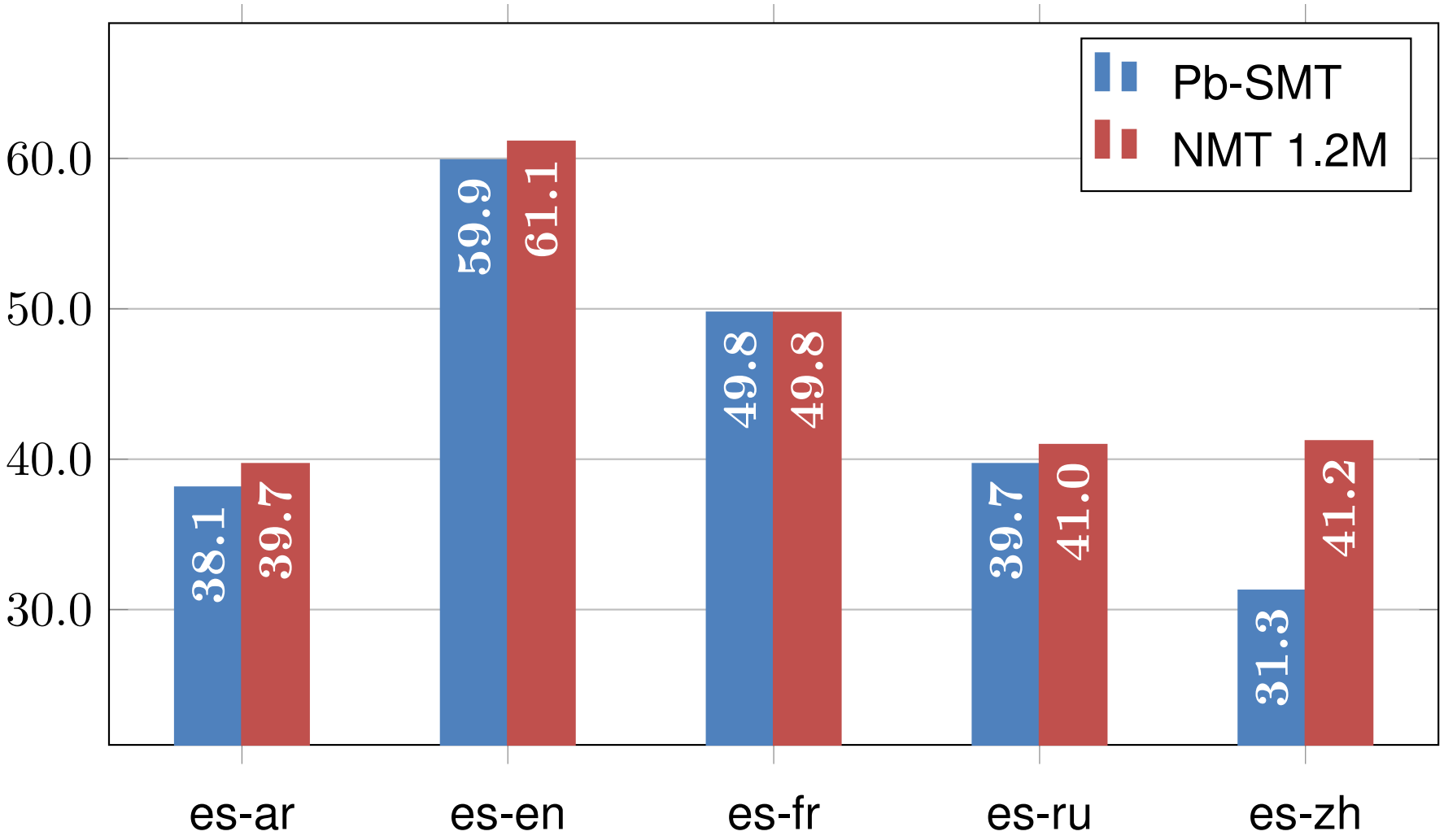
PB-SMT vs. NMT

Out of Spanish



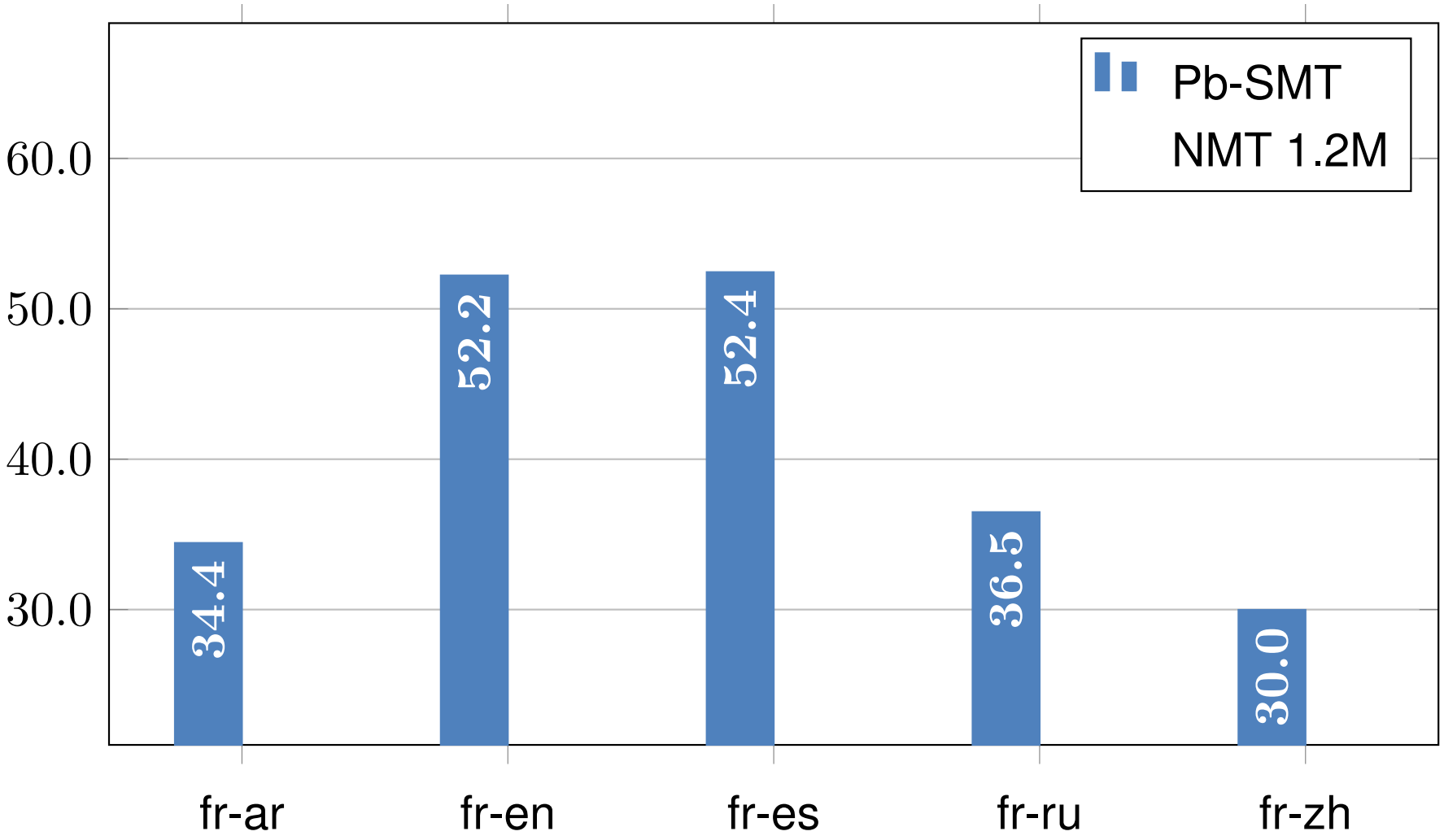
PB-SMT vs. NMT

Out of Spanish



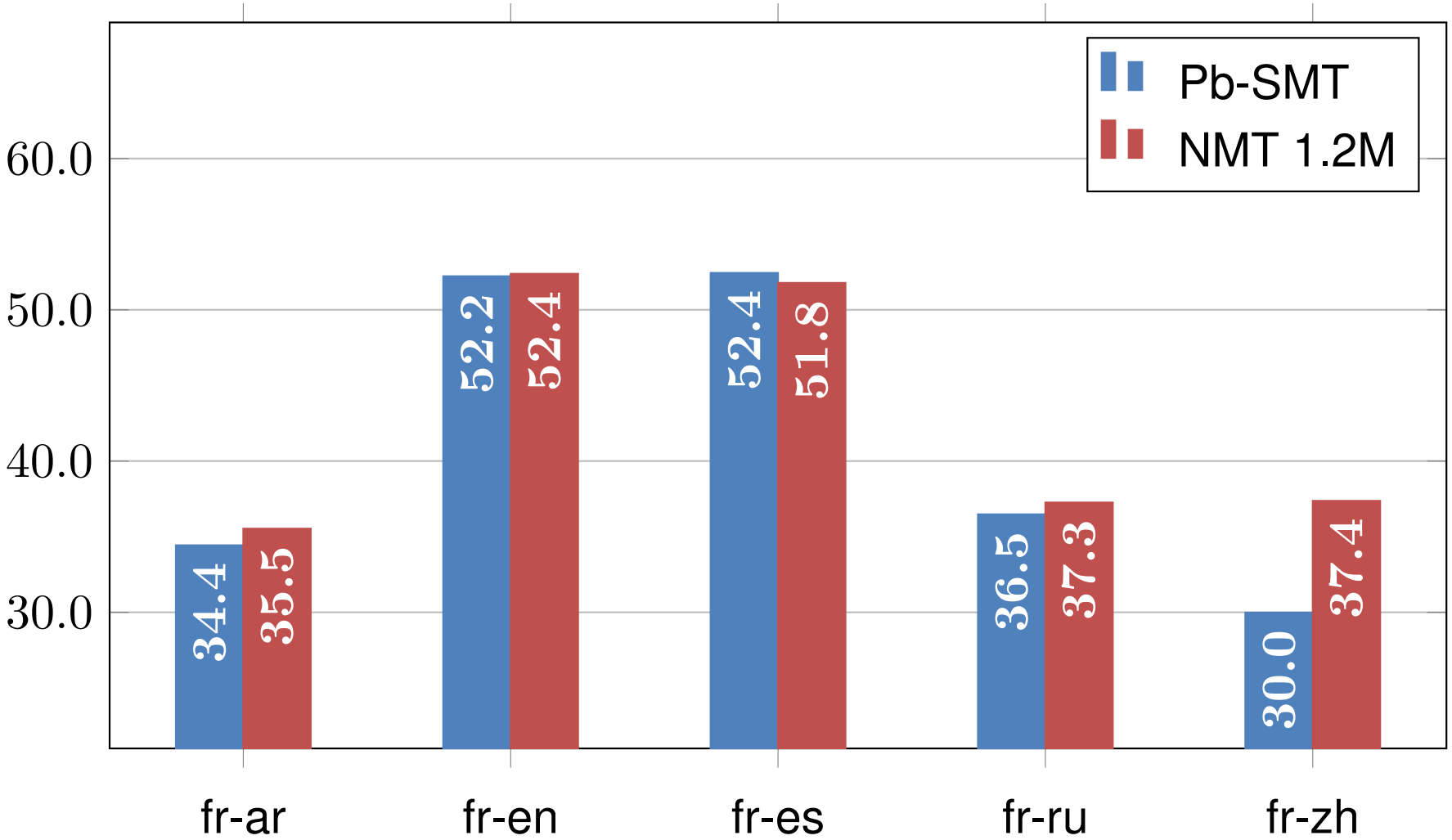
PB-SMT vs. NMT

Out of French



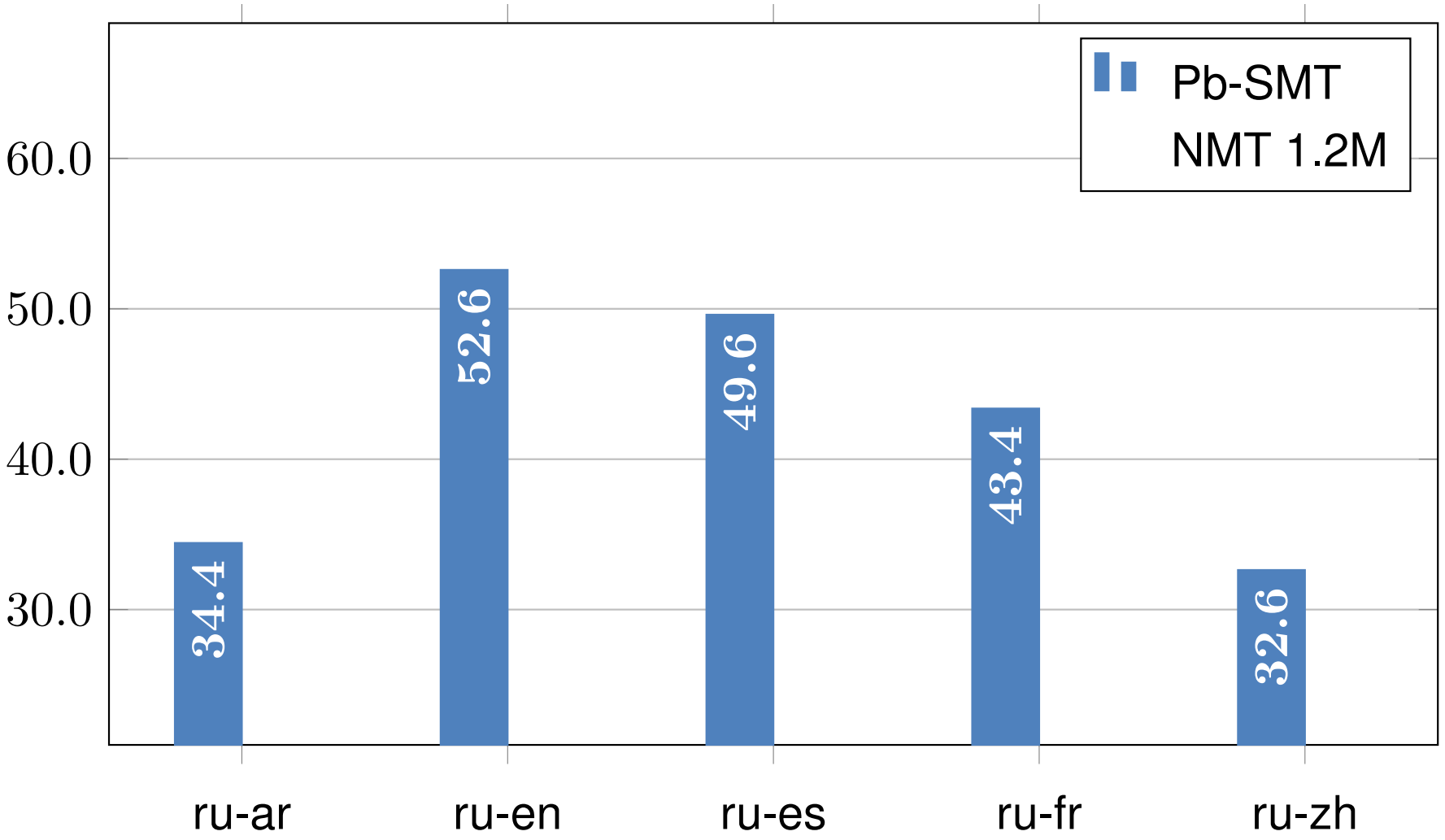
PB-SMT vs. NMT

Out of French



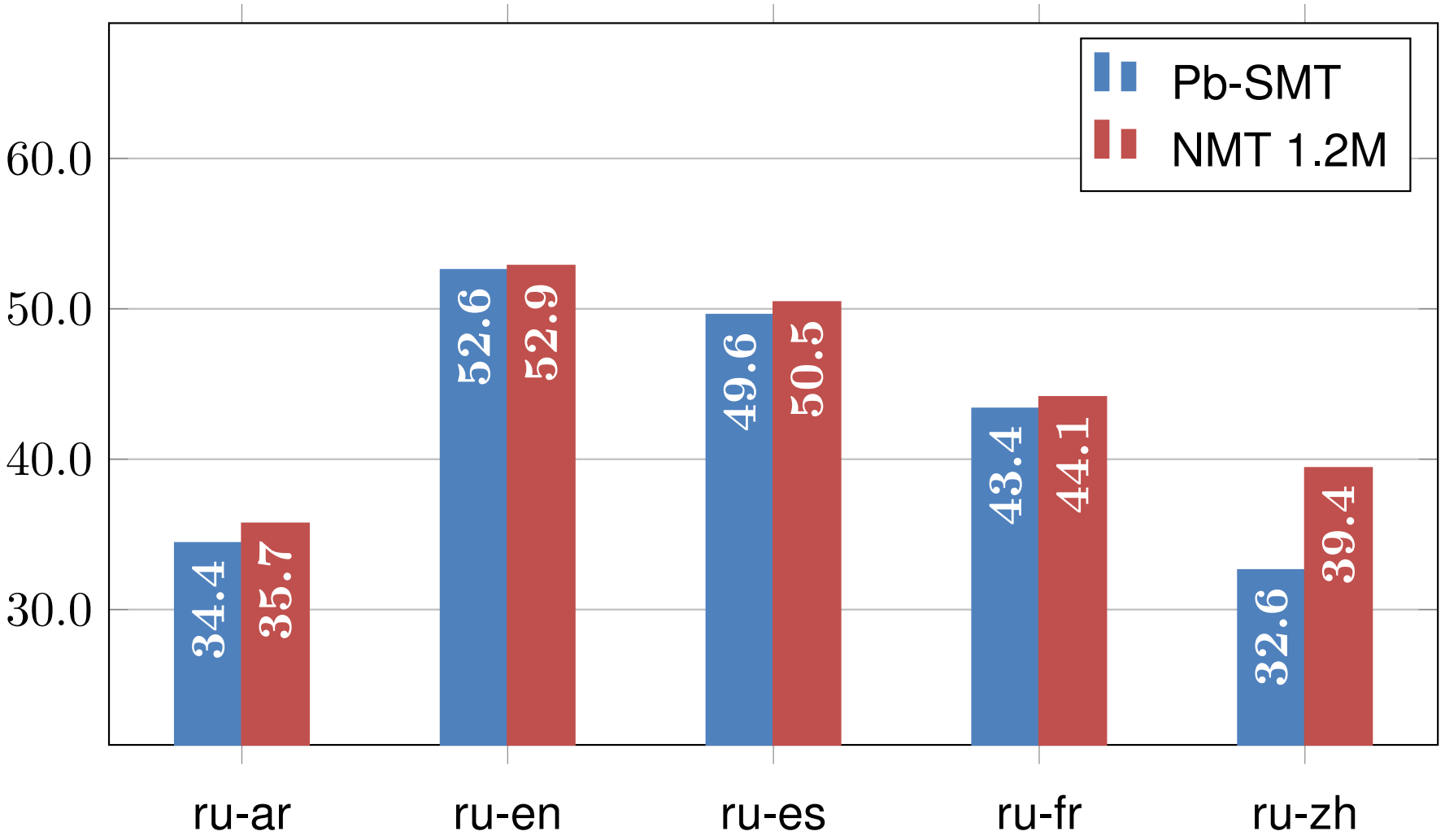
PB-SMT vs. NMT

Out of Russian



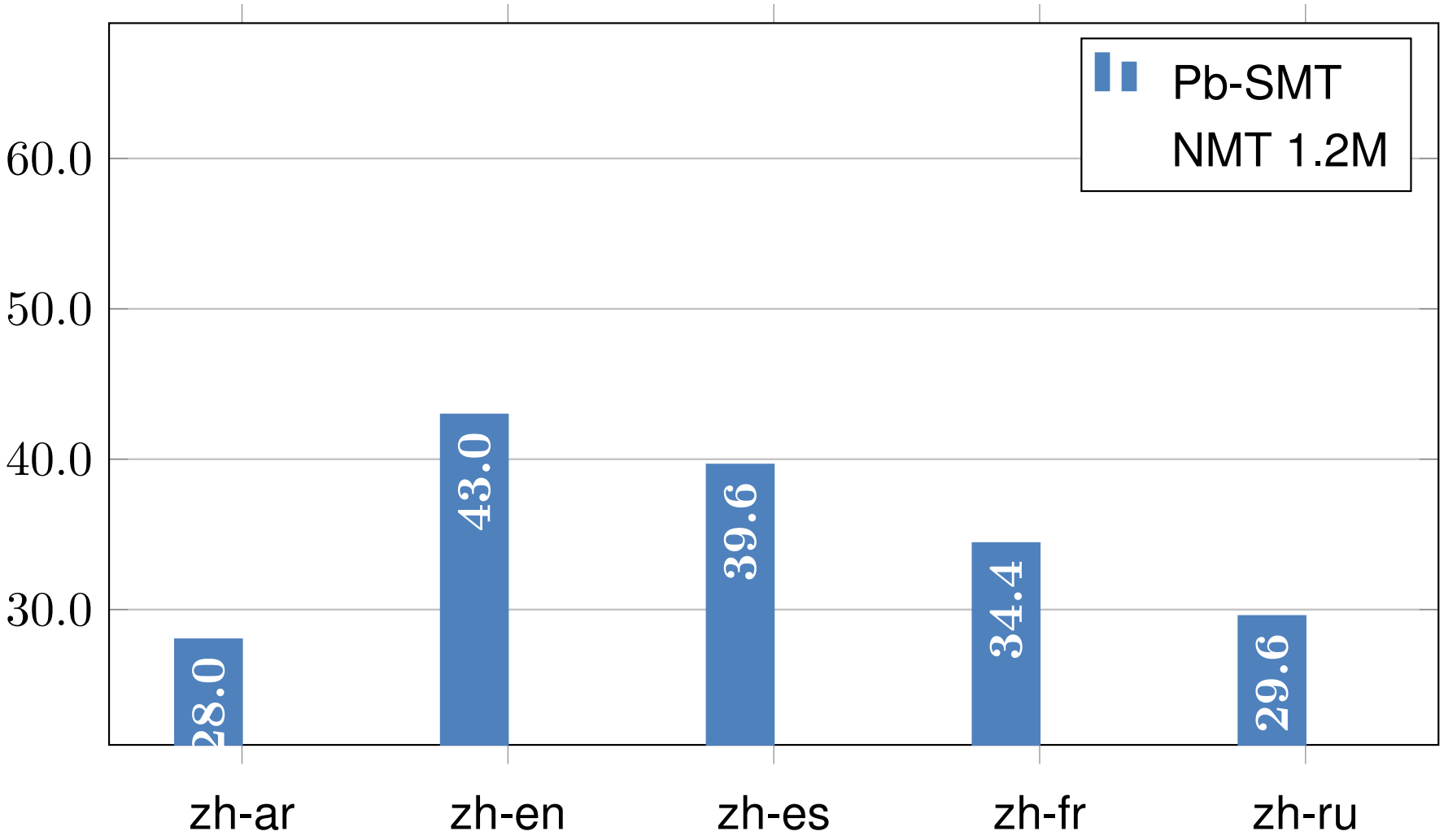
PB-SMT vs. NMT

Out of Russian



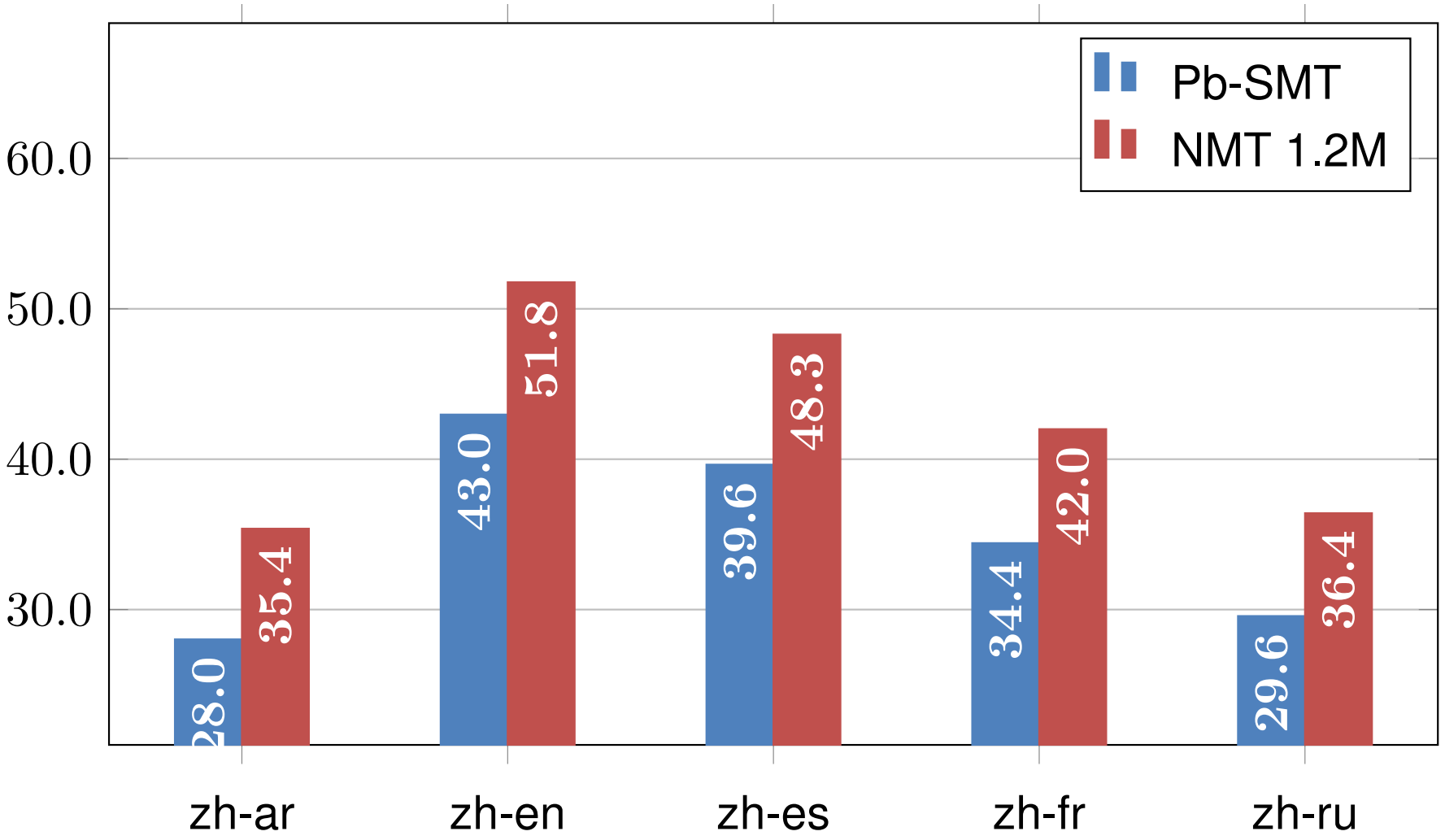
PB-SMT vs. NMT

Out of Chinese

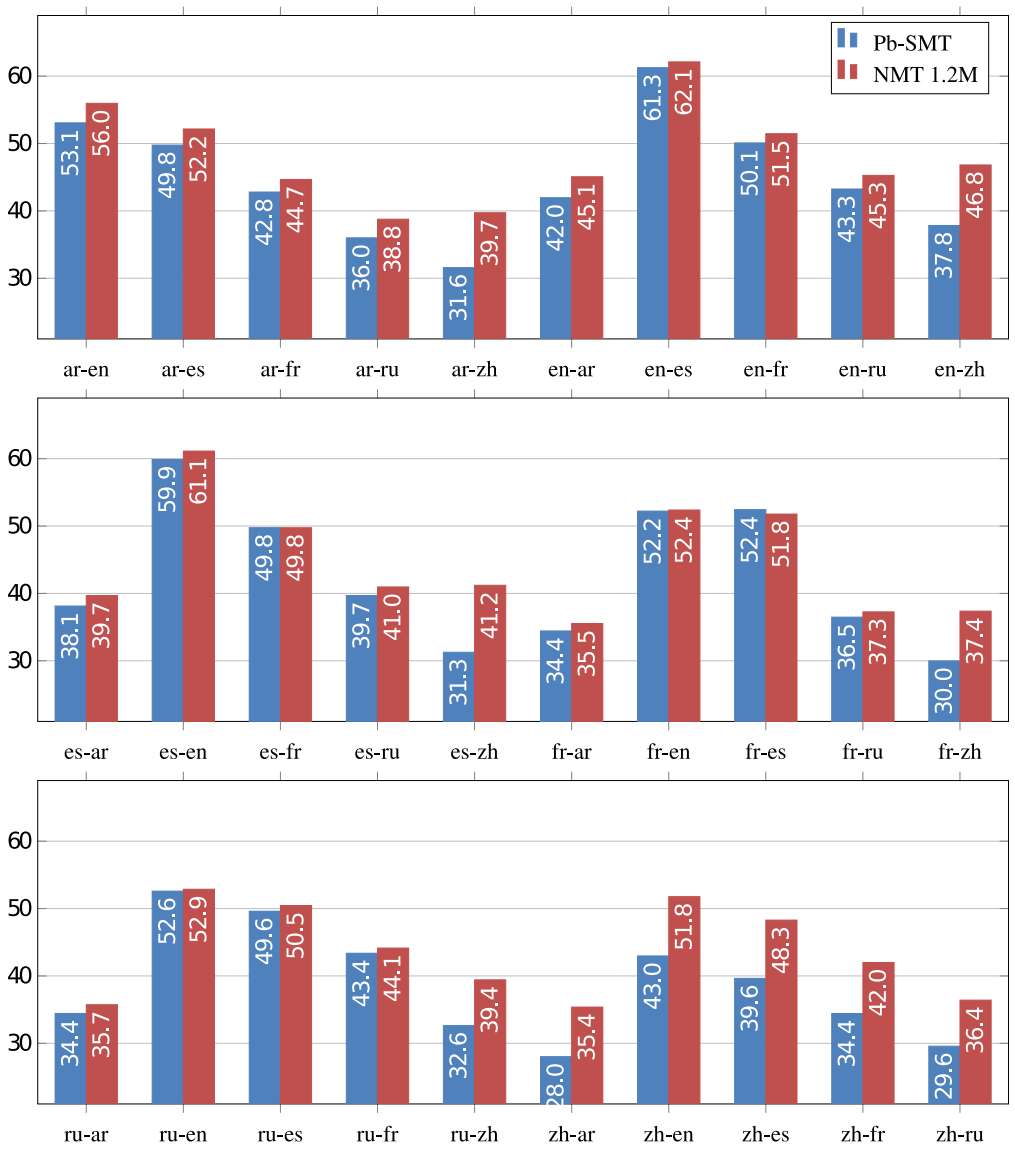


PB-SMT vs. NMT

Out of Chinese

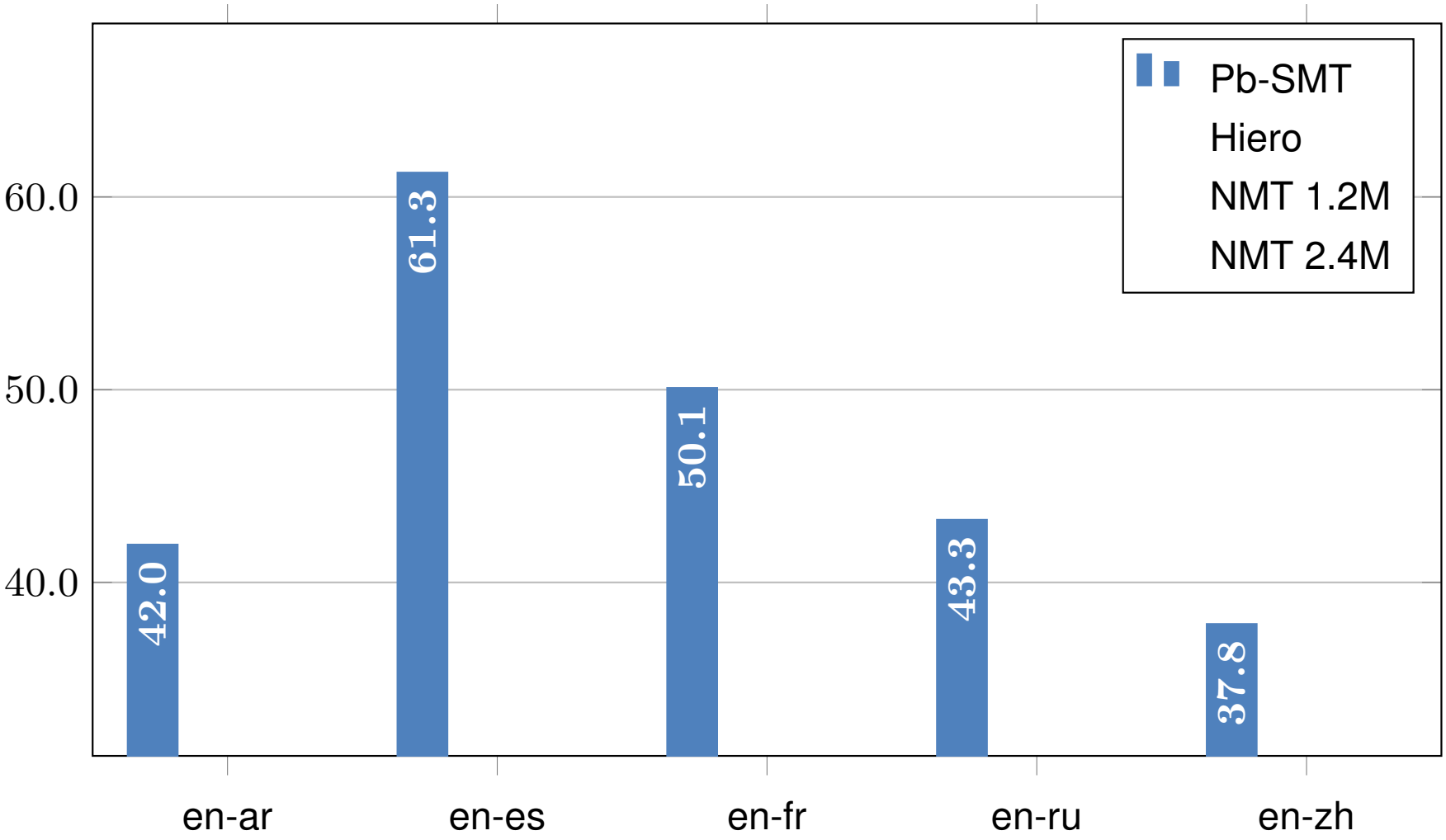


PB-SMT vs. NMT


Pb-SMT
NMT 1.2M
60
50
40
30
53.1 56.0
49.8 52.2
42.8 44.7
36.0 38.8
31.6 39.7
42.0 45.1
61.3 62.1
50.1 51.5
43.3 45.3
37.8 46.8
ar-en ar-es ar-fr ar-ru ar-zh en-ar en-es en-fr en-ru en-zh
38.1 39.7
59.9 61.1
49.8 49.8
39.7 41.0
31.3 41.2
34.4 35.5
52.2 52.4
52.4 51.8
36.5 37.3
30.0 37.4
es-ar es-en es-fr es-ru es-zh fr-ar fr-en fr-es fr-ru fr-zh
34.4 35.7
52.6 52.9
49.6 50.5
43.4 44.1
32.6 39.4
28.0 35.4
43.0 51.8
39.6 48.3
34.4 42.0
29.6 36.4
ru-ar ru-en ru-es ru-fr ru-zh zh-ar zh-en zh-es zh-fr zh-ru

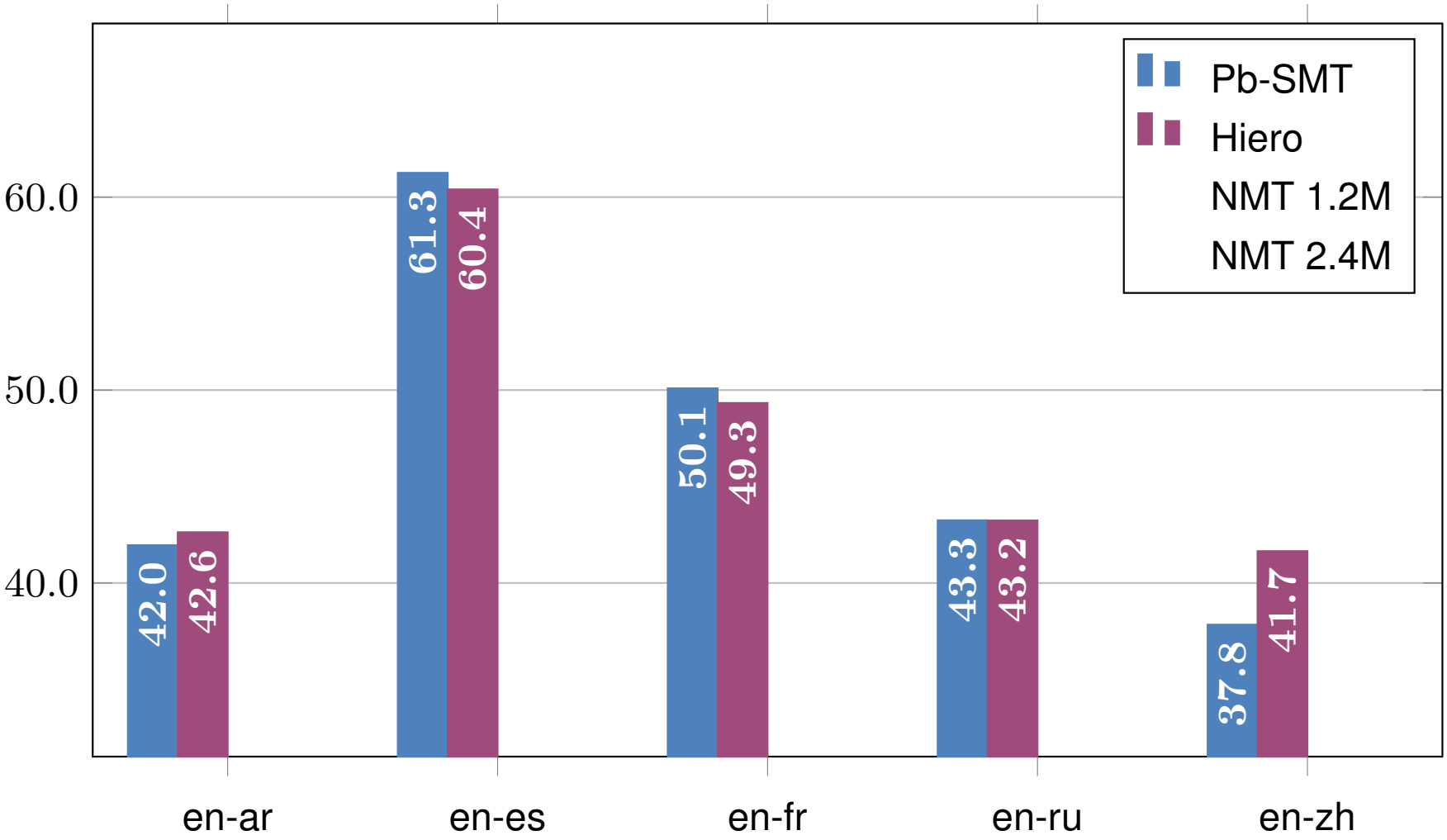
PB-SMT vs. Hiero vs. NMT

Out of English



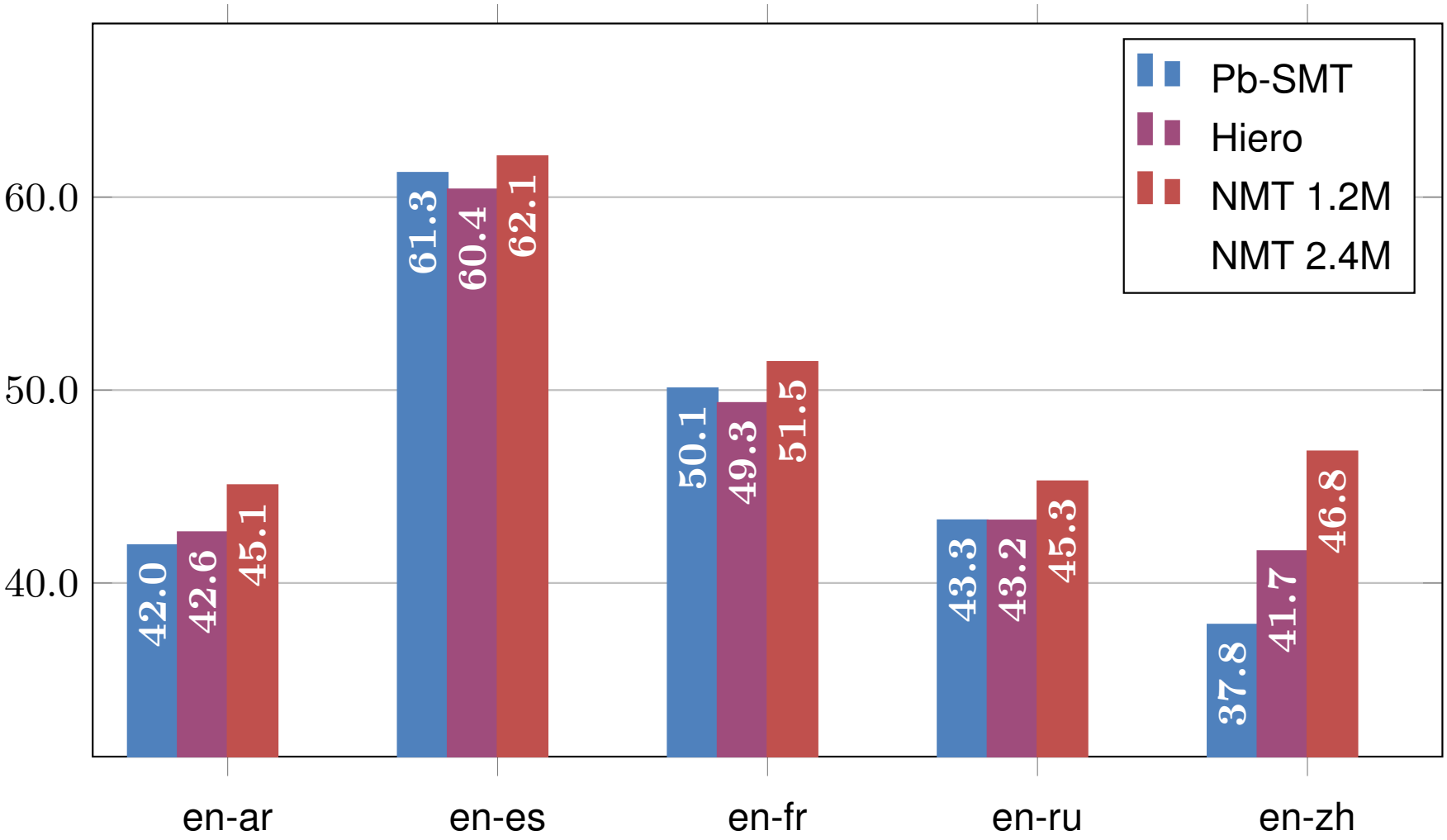
PB-SMT vs. Hiero vs. NMT

Out of English



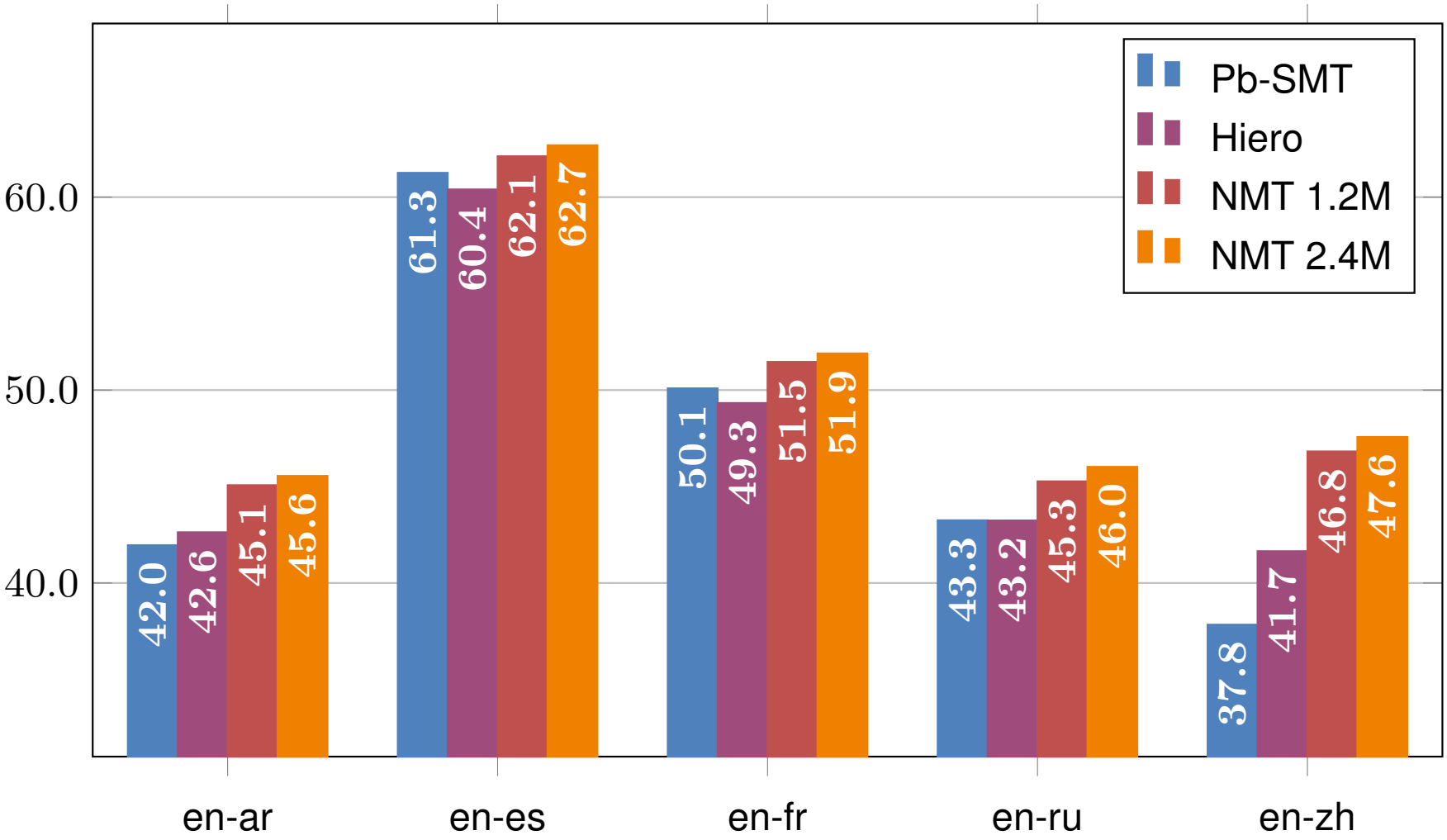
PB-SMT vs. Hiero vs. NMT

Out of English



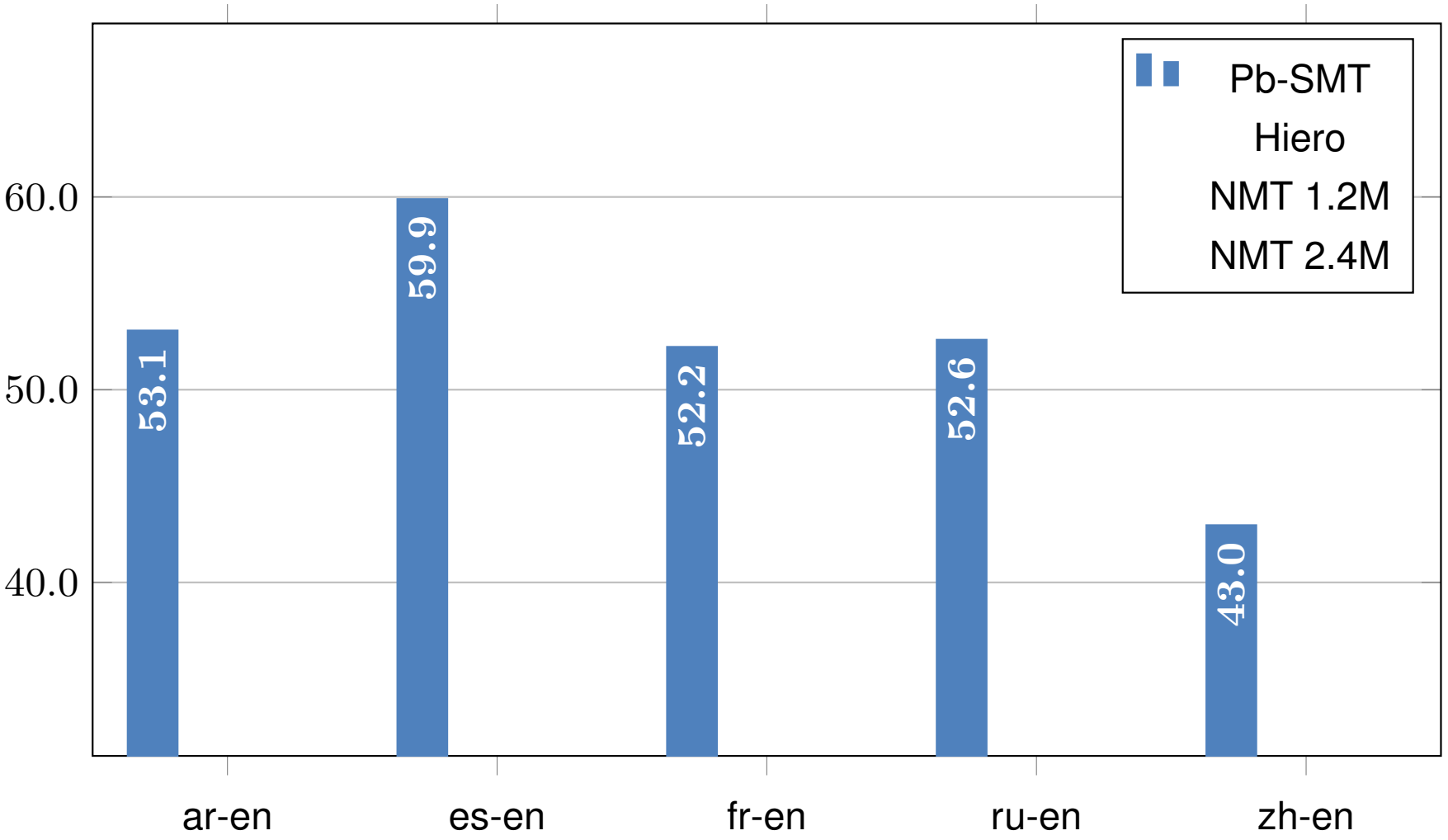
PB-SMT vs. Hiero vs. NMT

Out of English



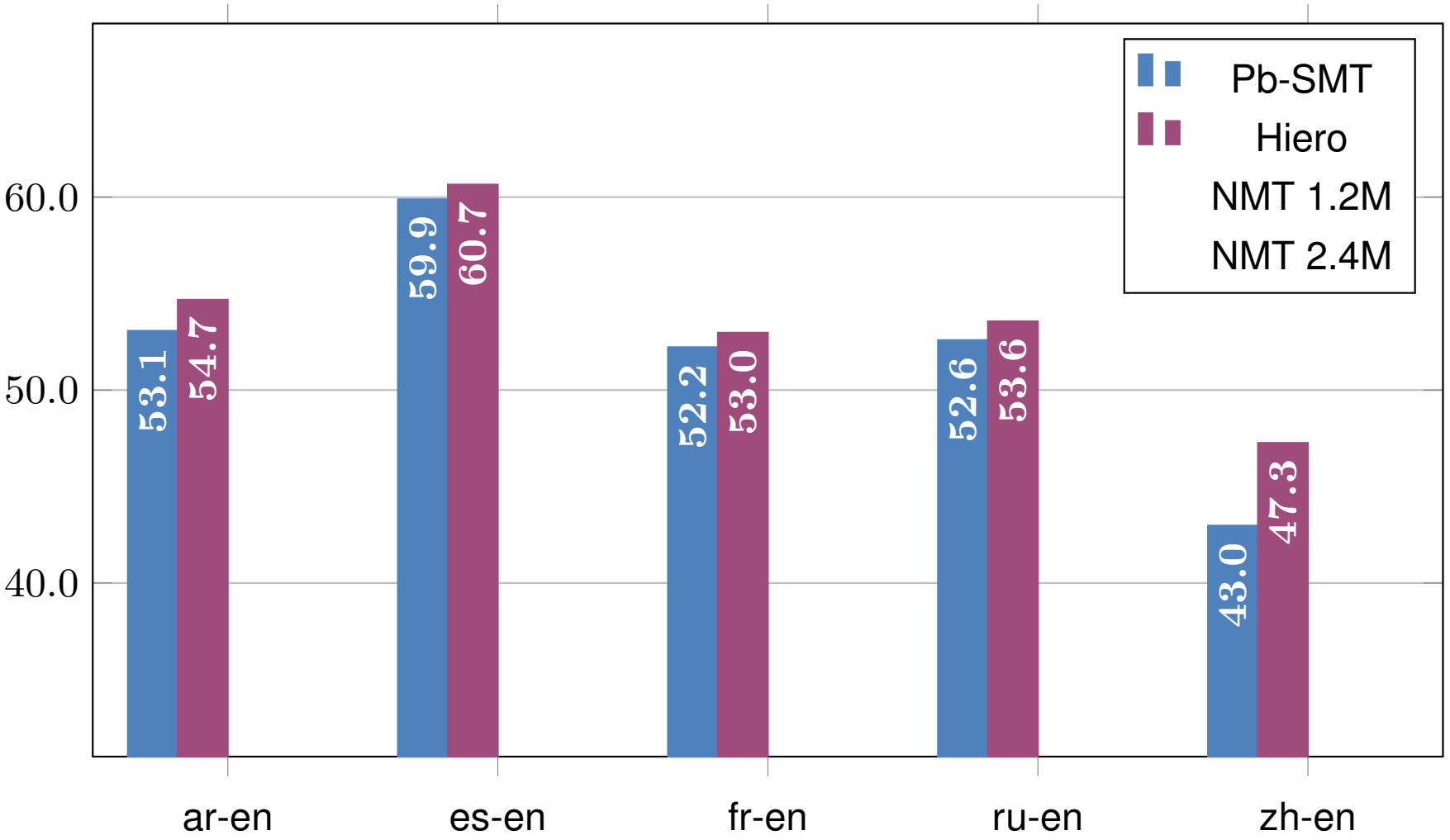
PB-SMT vs. Hiero vs. NMT

Into English



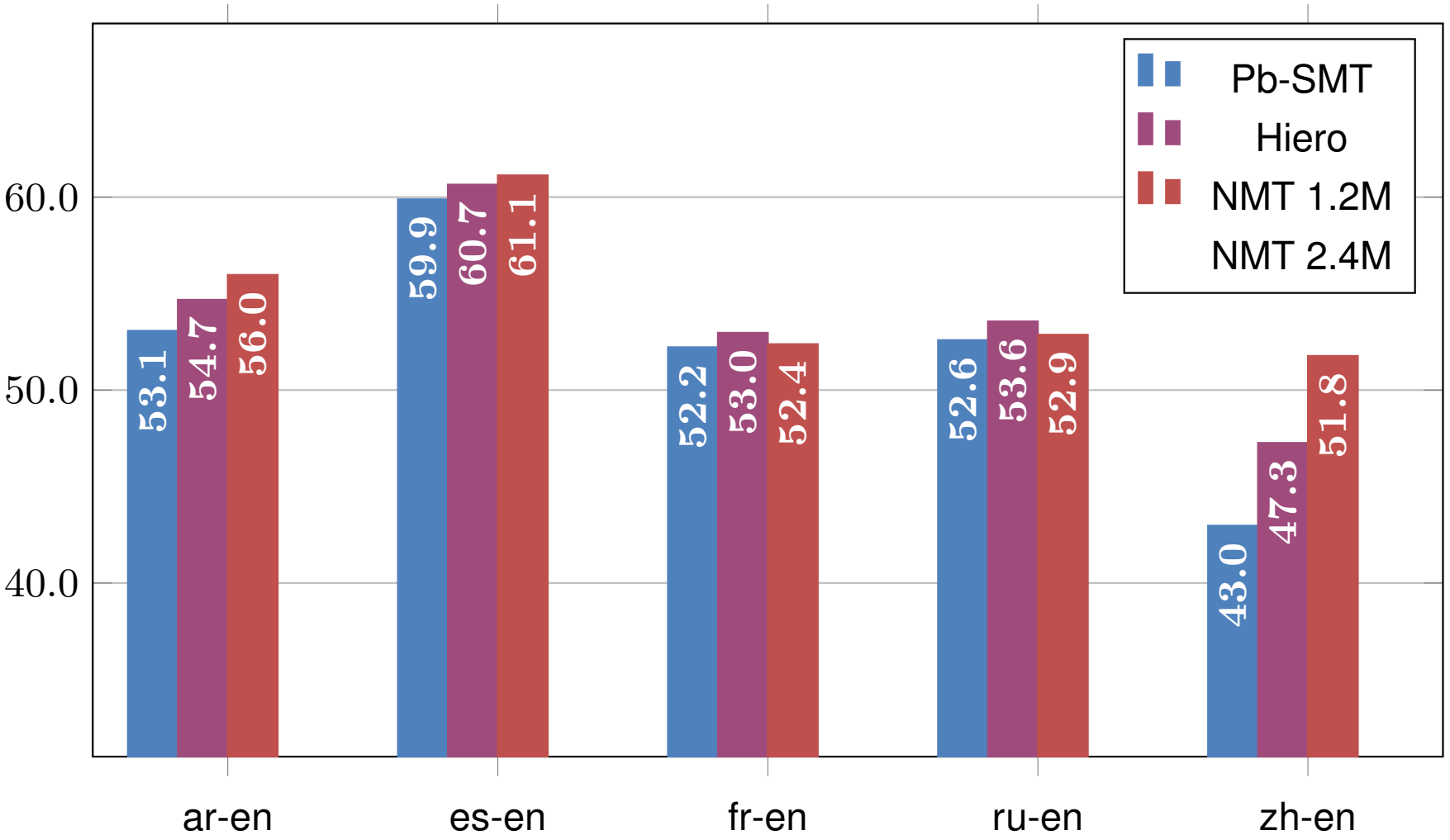
PB-SMT vs. Hiero vs. NMT

Into English



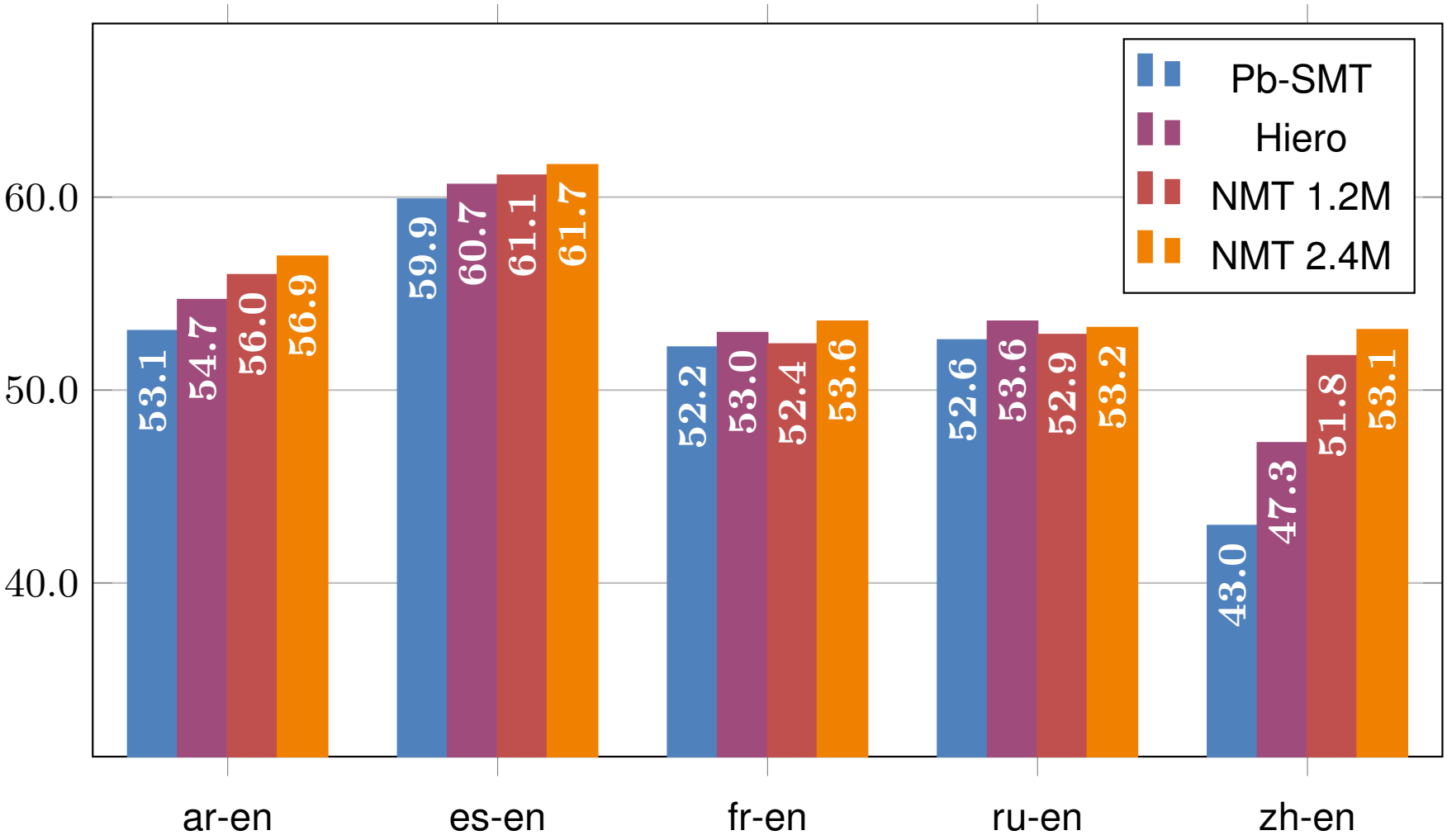
PB-SMT vs. Hiero vs. NMT

Into English



PB-SMT vs. Hiero vs. NMT

Into English



Is neural machine translation ready for deployment?

1. Is quality a problem?

2. Is translation speed a problem?

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Efficient decoding with AmuNMT

- AmuNMT[2] is a ground-up neural MT toolkit implementation, developed in C++;
- Grew out of a Moses feature function;
- Currently only inference (training framework in preparation);
- Compatible with Nematus [Sennrich et al., 2016];
- Multi-GPU, multi-CPU and mixed GPU/CPU mode with sentence-wise threads;
- Low-latency CPU-based decoding with intra-sentence multi-threading;
- Ensembling of multiple models;
- Vocabulary selection [Jean et al., 2015, Mi et al., 2016];
- Integrated segmentation into subwords [Sennrich et al., 2015].

[2] `https://github.com/emjotde/amunmt`

Checkpoint ensembling and averaging

System	BLEU	Wps	Memory
Single best	49.68	ca. 860	301M
Last-4 ensemble	51.38	ca. 200	1.2G
Last-4 average	51.13	ca. 860	301M
Last-8 average	51.35	ca. 860	301M

Table: Example for en-fr system

- Averaging models from different training runs does not work!

Checkpoint ensembling and averaging

System	BLEU	Wps	Memory
Single best	49.68	ca. 860	301M
Last-4 ensemble	51.38	ca. 200	1.2G
Last-4 average	51.13	ca. 860	301M
Last-8 average	51.35	ca. 860	301M

Table: Example for en-fr system

- Averaging models from different training runs does not work!

Checkpoint ensembling and averaging

System	BLEU	Wps	Memory
Single best	49.68	ca. 860	301M
Last-4 ensemble	51.38	ca. 200	1.2G
Last-4 average	51.13	ca. 860	301M
Last-8 average	51.35	ca. 860	301M

Table: Example for en-fr system

- Averaging models from different training runs does not work!

Checkpoint ensembling and averaging

System	BLEU	Wps	Memory
Single best	49.68	ca. 860	301M
Last-4 ensemble	51.38	ca. 200	1.2G
Last-4 average	51.13	ca. 860	301M
Last-8 average	51.35	ca. 860	301M

Table: Example for en-fr system

- Averaging models from different training runs does not work!

Vocabulary selection

- Decoding time on CPU is dominated by the final output layer (30,000 vocabulary items);
- [Jean et al., 2015] proposed to clip the per-sentence vocabulary to the $K = 30$k (out of 500k) most common target words and $K' = 10$ most probable translations for each source word.
- We find that choosing $K = 75$ and $K' = 75$ does not hurt quality (similar as [Mi et al., 2016]);
- Results on average in ca. 1250 vocabulary items per sentence.
- BLEU scores remain the same for slightly changed translation.

Vocabulary selection

- Decoding time on CPU is dominated by the final output layer (30,000 vocabulary items);
- [Jean et al., 2015] proposed to clip the per-sentence vocabulary to the $K = 30$k (out of 500k) most common target words and $K' = 10$ most probable translations for each source word.
- We find that choosing $K = 75$ and $K' = 75$ does not hurt quality (similar as [Mi et al., 2016]);
- Results on average in ca. 1250 vocabulary items per sentence.
- BLEU scores remain the same for slightly changed translation.

Vocabulary selection

- Decoding time on CPU is dominated by the final output layer (30,000 vocabulary items);
- [Jean et al., 2015] proposed to clip the per-sentence vocabulary to the $K = 30$k (out of 500k) most common target words and $K' = 10$ most probable translations for each source word.
- We find that choosing $K = 75$ and $K' = 75$ does not hurt quality (similar as [Mi et al., 2016]);
- Results on average in ca. 1250 vocabulary items per sentence.
- BLEU scores remain the same for slightly changed translation.

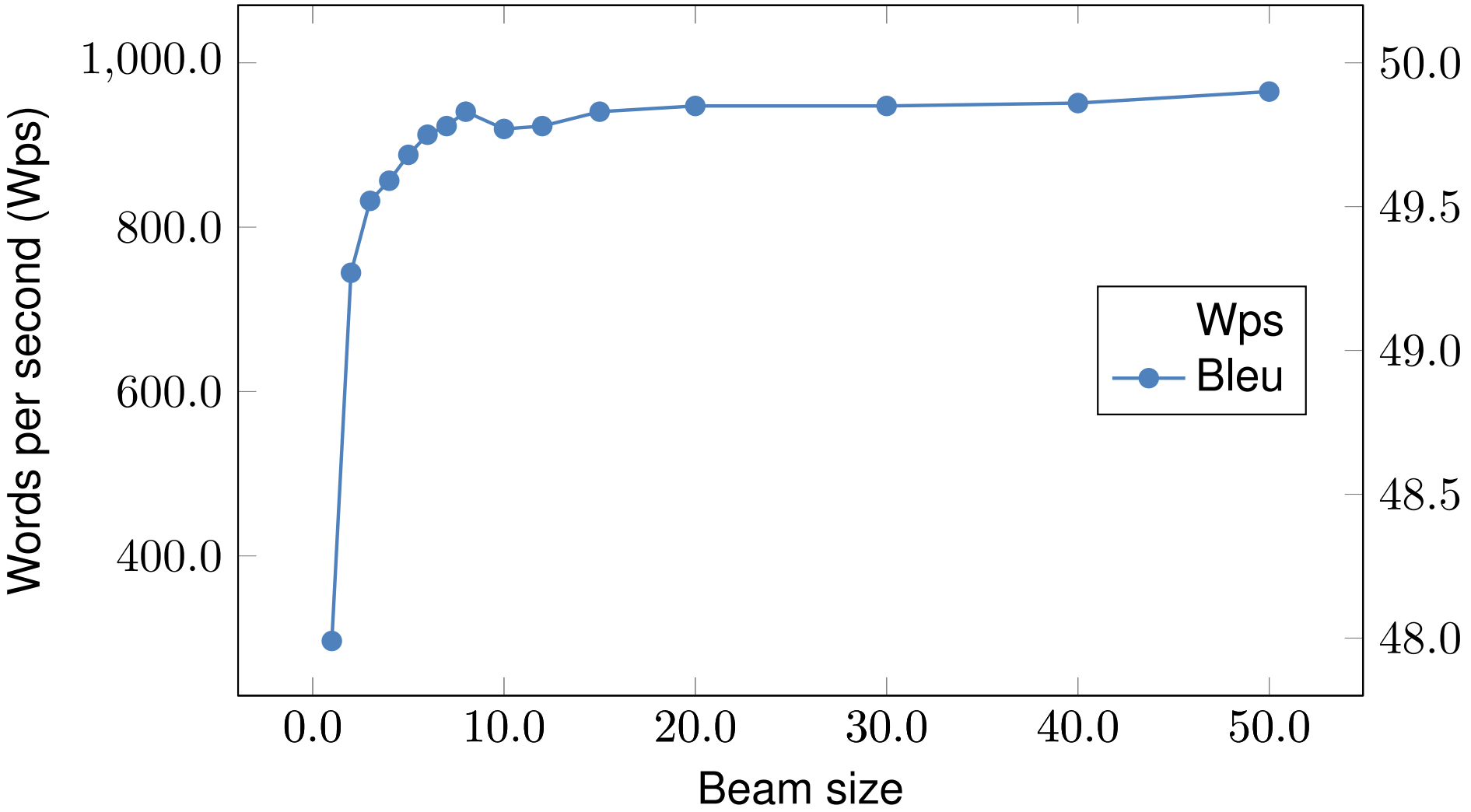
Vocabulary selection

- Decoding time on CPU is dominated by the final output layer (30,000 vocabulary items);
- [Jean et al., 2015] proposed to clip the per-sentence vocabulary to the $K = 30$k (out of 500k) most common target words and $K' = 10$ most probable translations for each source word.
- We find that choosing $K = 75$ and $K' = 75$ does not hurt quality (similar as [Mi et al., 2016]);
- Results on average in ca. 1250 vocabulary items per sentence.
- BLEU scores remain the same for slightly changed translation.

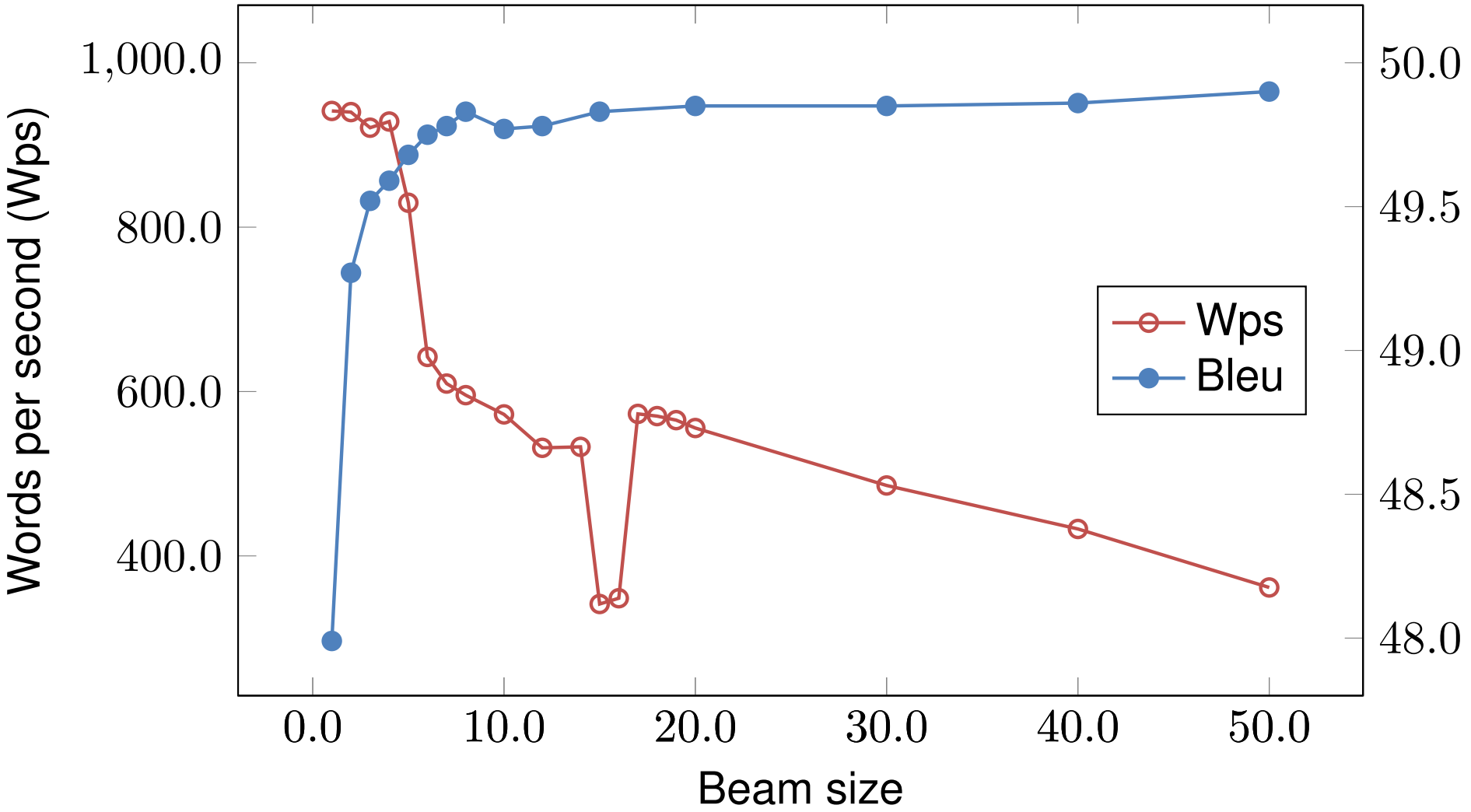
Vocabulary selection

- Decoding time on CPU is dominated by the final output layer (30,000 vocabulary items);
- [Jean et al., 2015] proposed to clip the per-sentence vocabulary to the $K = 30$k (out of 500k) most common target words and $K' = 10$ most probable translations for each source word.
- We find that choosing $K = 75$ and $K' = 75$ does not hurt quality (similar as [Mi et al., 2016]);
- Results on average in ca. 1250 vocabulary items per sentence.
- BLEU scores remain the same for slightly changed translation.

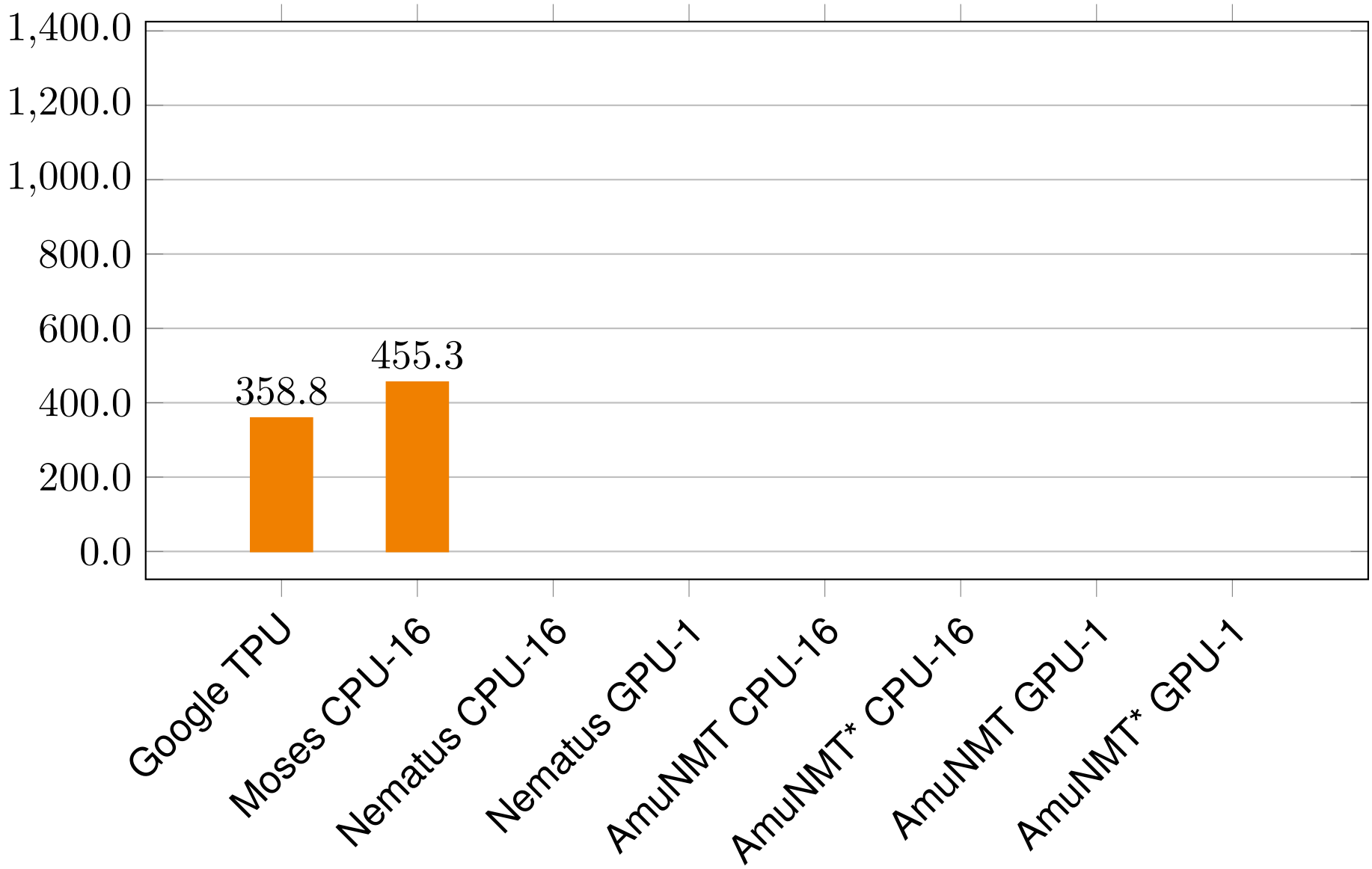
Speed and quality vs. beam size



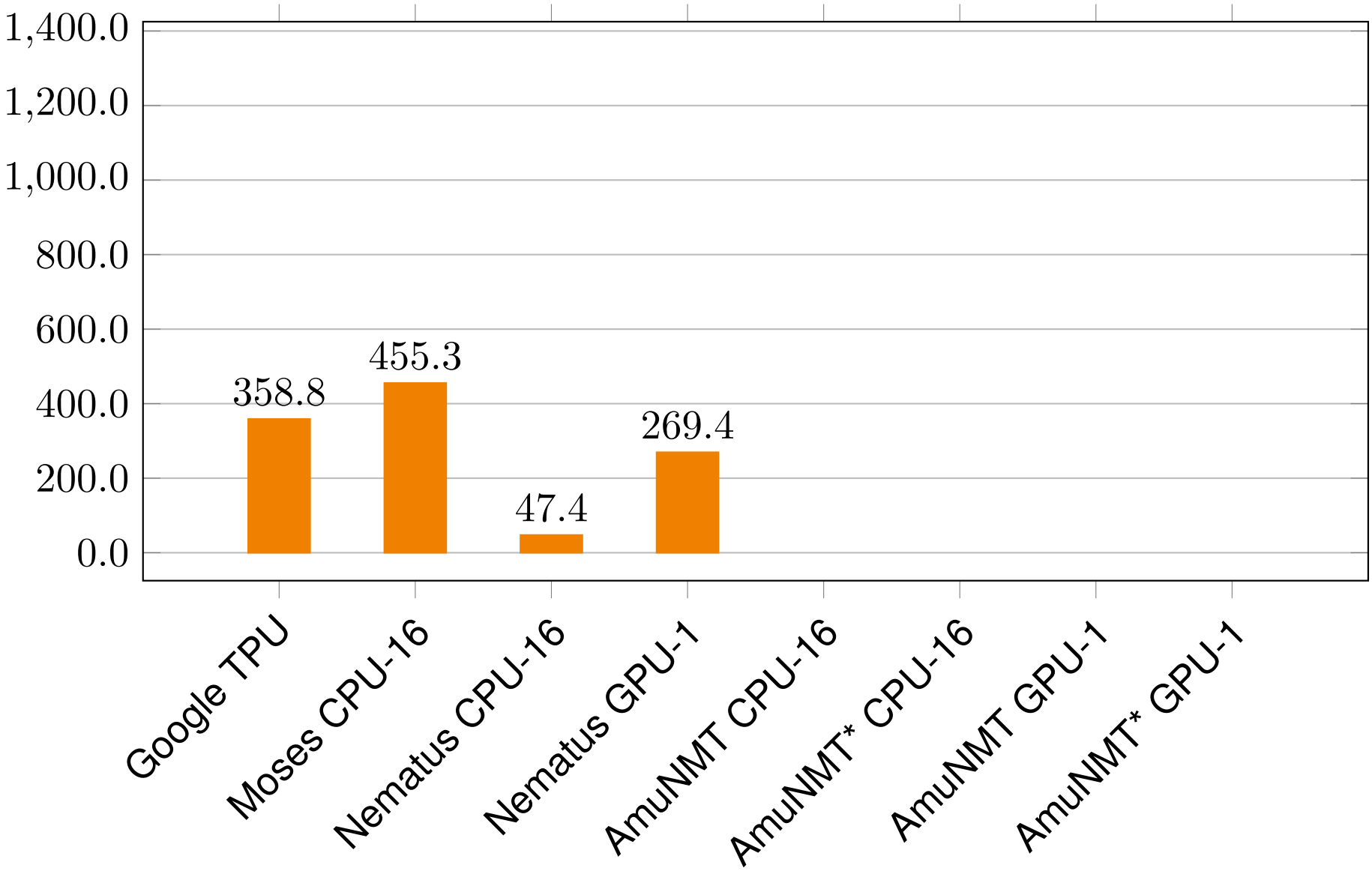
Speed and quality vs. beam size



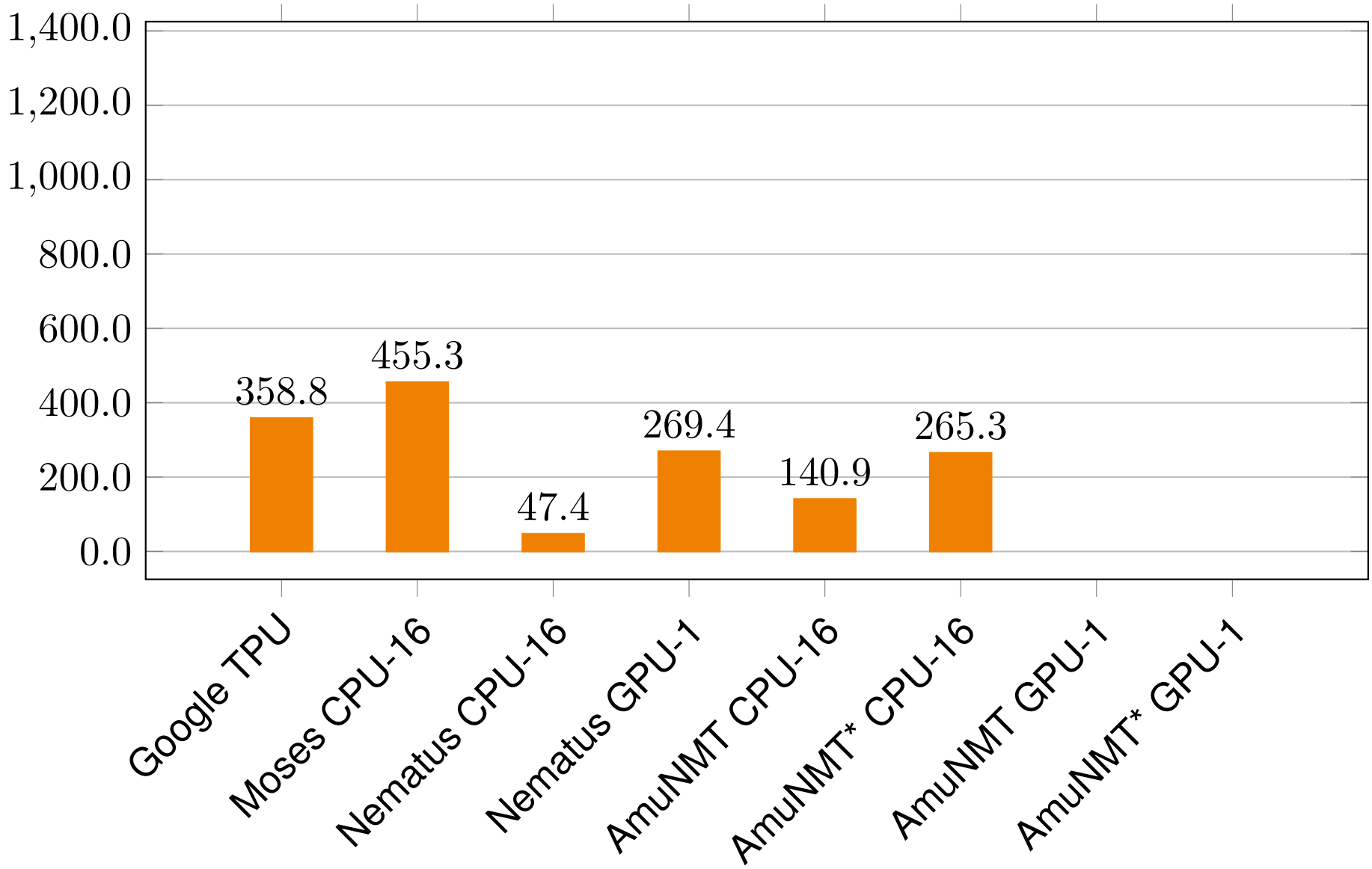
Translation speed in words per second



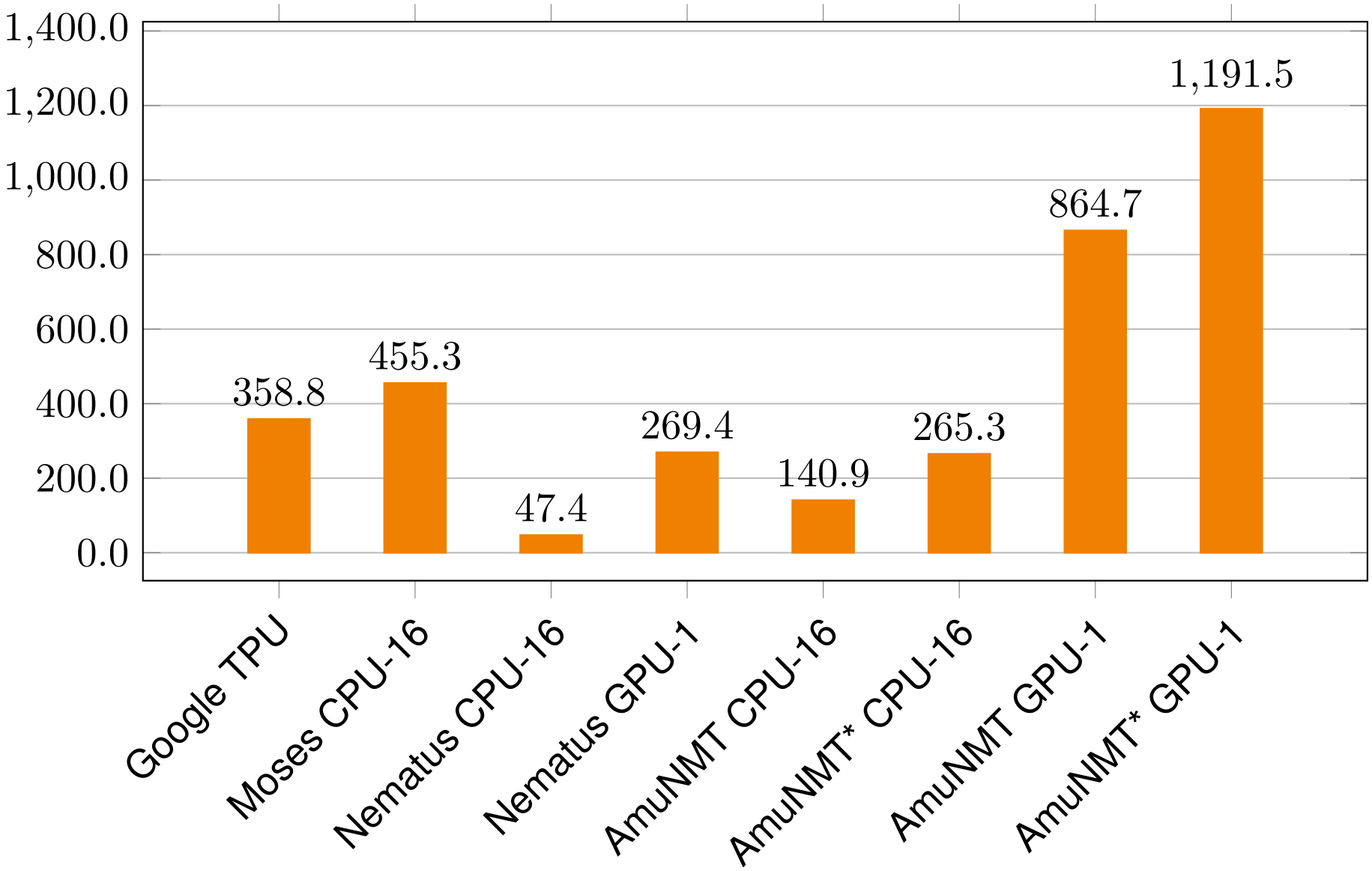
Translation speed in words per second



Translation speed in words per second

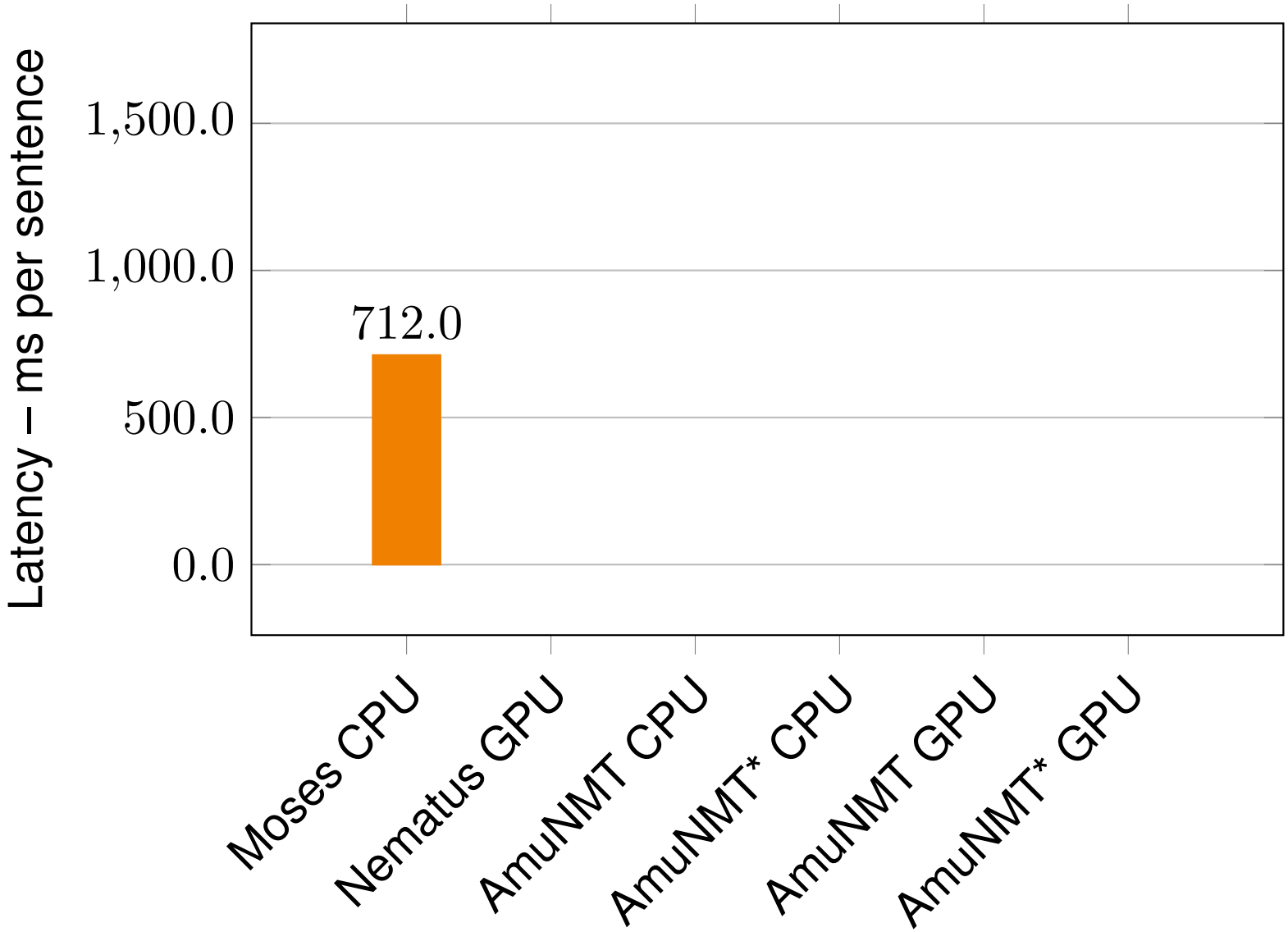


Translation speed in words per second



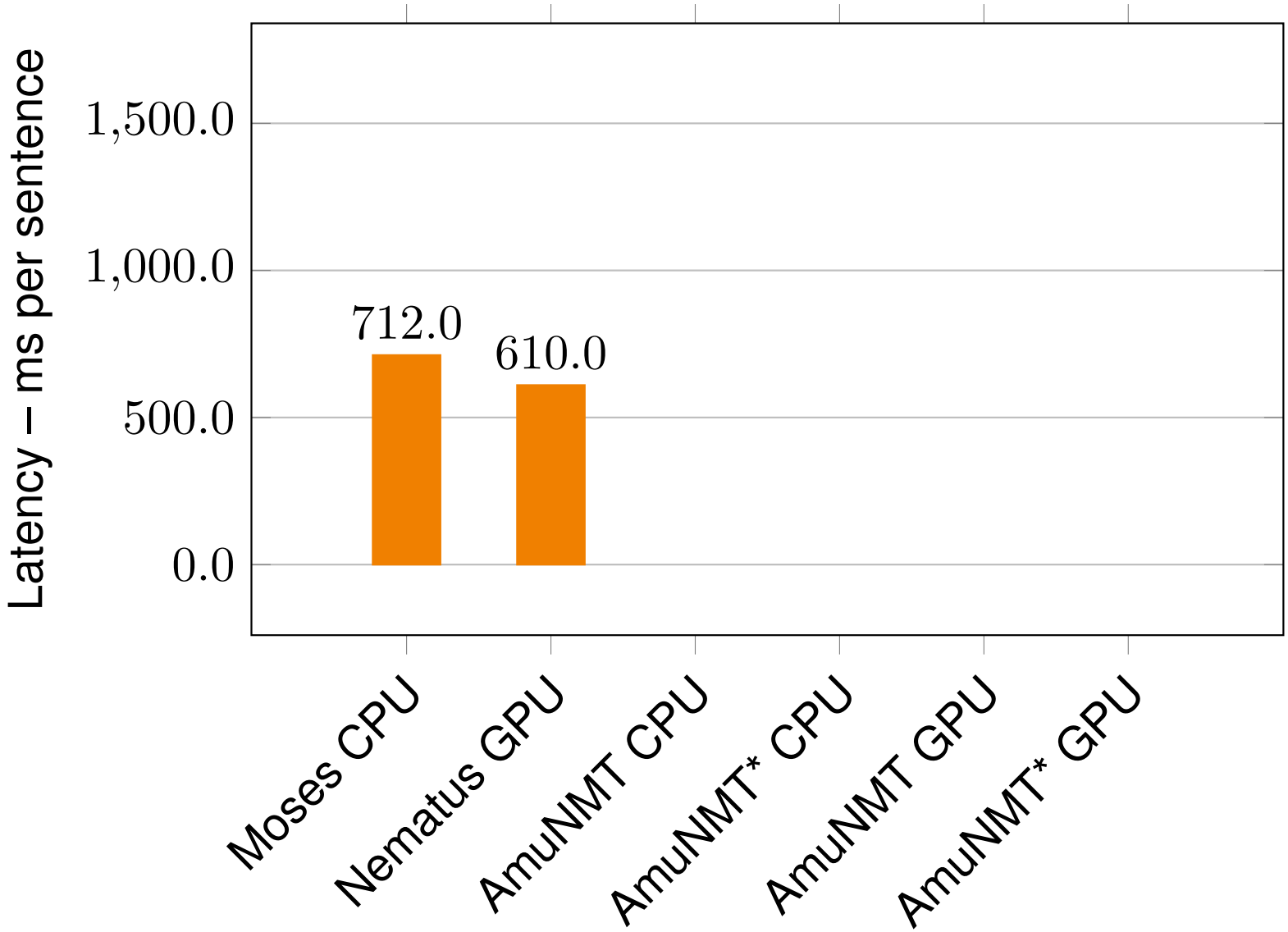
Latency in ms per sentence

Lower is better



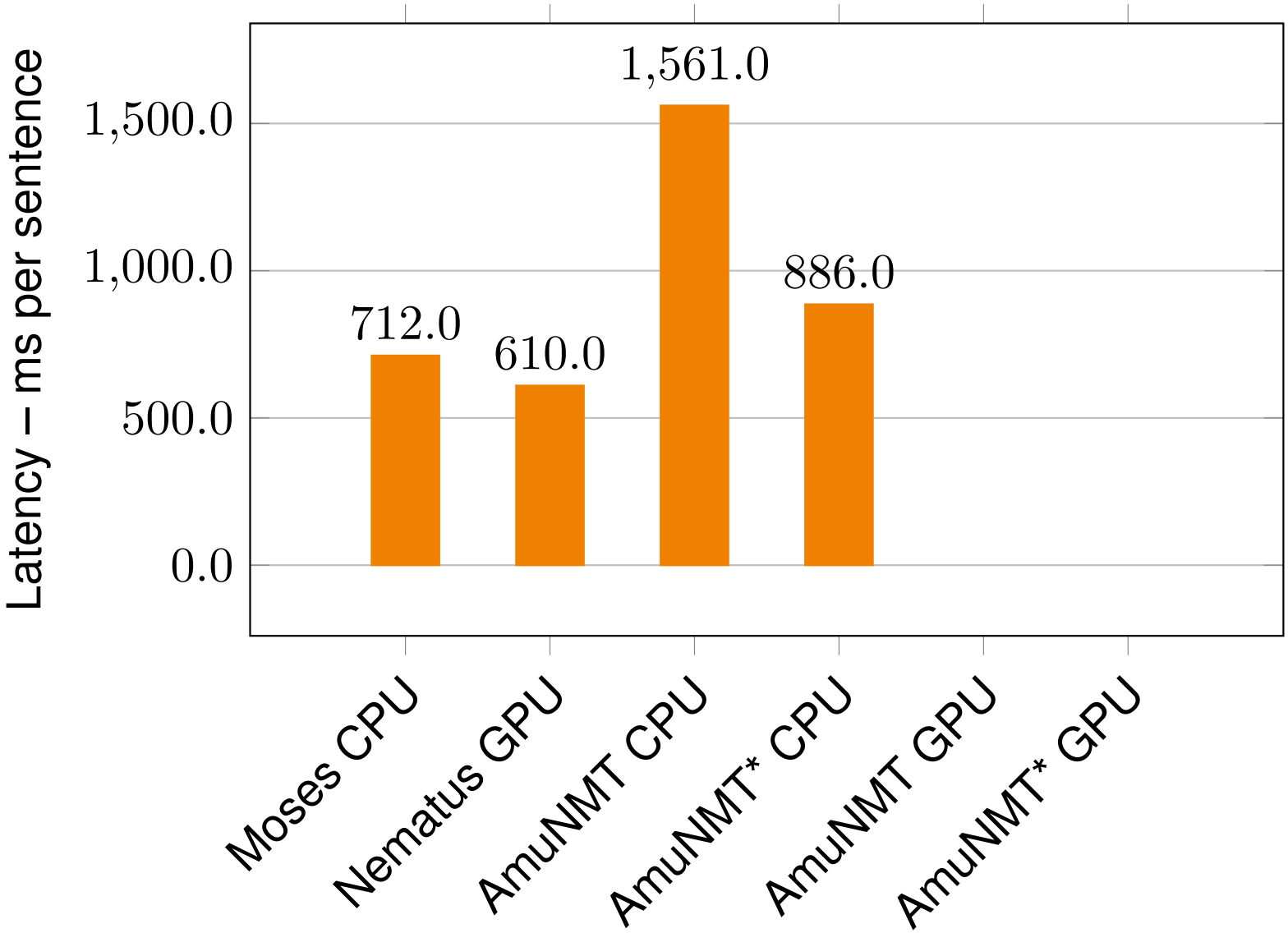
Latency in ms per sentence

Lower is better



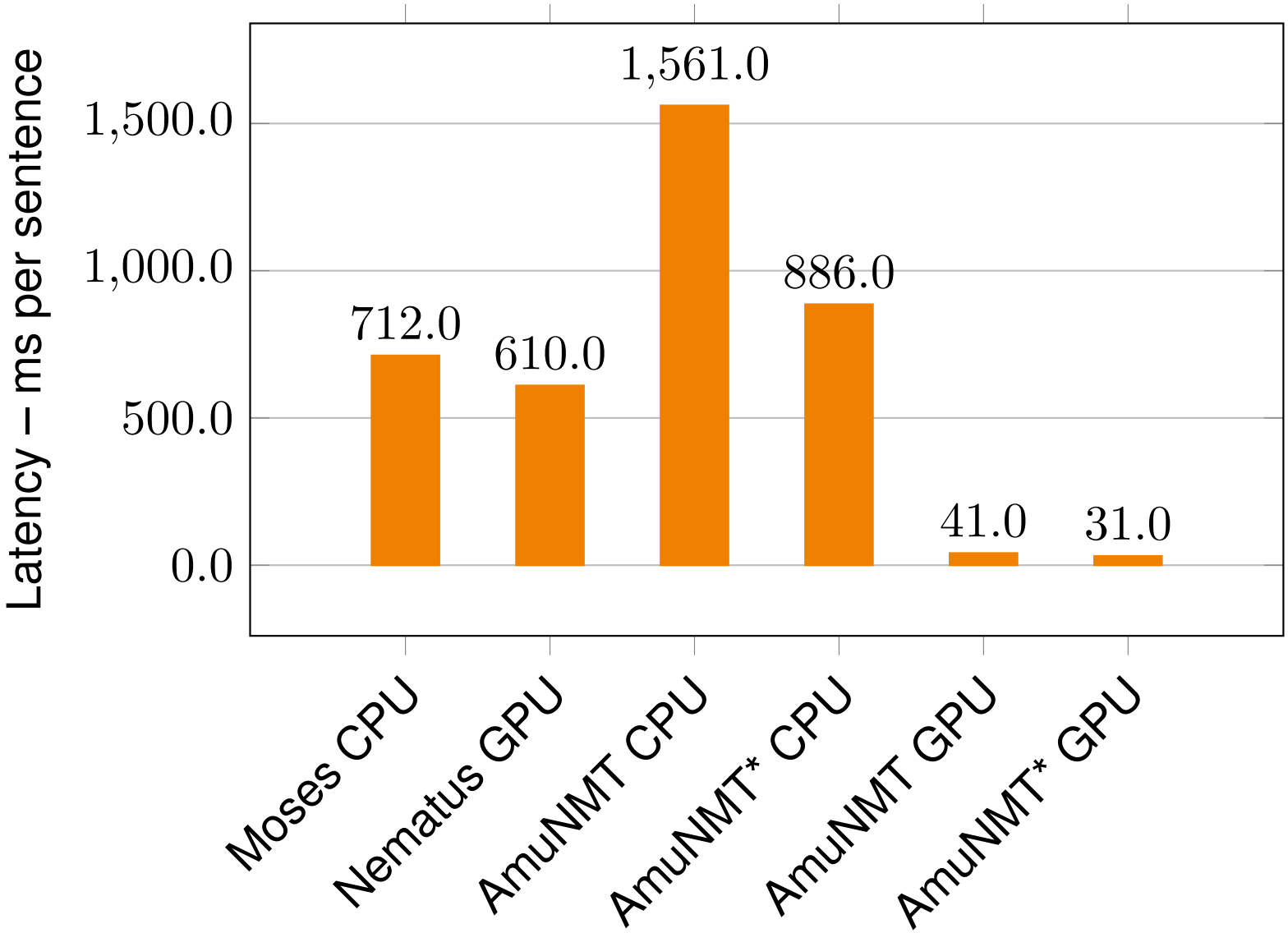
Latency in ms per sentence

Lower is better



Latency in ms per sentence

Lower is better



Conclusions

- Replacing PB-SMT with NMT seems to incurs no quality loss in terms of BLEU, in most cases improvements;
- Here: quality after 10 days of training mostly converged, small gains still possible;
- NMT (WMT-grade models) can be deployed on commodity hardware;
- CPU-bound translation at similar speed as phrase-based SMT for in-house setup;
- GPU-bound translation much faster;
- Future work: determine impact of model size, hidden layers on speed and quality;
- Future work: human evaluation in real-world setting.

Conclusions

- Replacing PB-SMT with NMT seems to incurs no quality loss in terms of BLEU, in most cases improvements;
- Here: quality after 10 days of training mostly converged, small gains still possible;
- NMT (WMT-grade models) can be deployed on commodity hardware;
- CPU-bound translation at similar speed as phrase-based SMT for in-house setup;
- GPU-bound translation much faster;
- Future work: determine impact of model size, hidden layers on speed and quality;
- Future work: human evaluation in real-world setting.

Conclusions

- Replacing PB-SMT with NMT seems to incurs no quality loss in terms of BLEU, in most cases improvements;
- Here: quality after 10 days of training mostly converged, small gains still possible;
- NMT (WMT-grade models) can be deployed on commodity hardware;
- CPU-bound translation at similar speed as phrase-based SMT for in-house setup;
- GPU-bound translation much faster;
- Future work: determine impact of model size, hidden layers on speed and quality;
- Future work: human evaluation in real-world setting.

Conclusions

- Replacing PB-SMT with NMT seems to incurs no quality loss in terms of BLEU, in most cases improvements;
- Here: quality after 10 days of training mostly converged, small gains still possible;
- NMT (WMT-grade models) can be deployed on commodity hardware;
- CPU-bound translation at similar speed as phrase-based SMT for in-house setup;
- GPU-bound translation much faster;
- Future work: determine impact of model size, hidden layers on speed and quality;
- Future work: human evaluation in real-world setting.

Conclusions

- Replacing PB-SMT with NMT seems to incurs no quality loss in terms of BLEU, in most cases improvements;
- Here: quality after 10 days of training mostly converged, small gains still possible;
- NMT (WMT-grade models) can be deployed on commodity hardware;
- CPU-bound translation at similar speed as phrase-based SMT for in-house setup;
- GPU-bound translation much faster;
- Future work: determine impact of model size, hidden layers on speed and quality;
- Future work: human evaluation in real-world setting.

Conclusions

- Replacing PB-SMT with NMT seems to incurs no quality loss in terms of BLEU, in most cases improvements;
- Here: quality after 10 days of training mostly converged, small gains still possible;
- NMT (WMT-grade models) can be deployed on commodity hardware;
- CPU-bound translation at similar speed as phrase-based SMT for in-house setup;
- GPU-bound translation much faster;
- Future work: determine impact of model size, hidden layers on speed and quality;
- Future work: human evaluation in real-world setting.

Conclusions

- Replacing PB-SMT with NMT seems to incurs no quality loss in terms of BLEU, in most cases improvements;
- Here: quality after 10 days of training mostly converged, small gains still possible;
- NMT (WMT-grade models) can be deployed on commodity hardware;
- CPU-bound translation at similar speed as phrase-based SMT for in-house setup;
- GPU-bound translation much faster;
- Future work: determine impact of model size, hidden layers on speed and quality;
- Future work: human evaluation in real-world setting.

Bonus slide I

Deployment at WIPO (World Intellectual Property Organization)

SLATOR
Language Industry Intelligence

NEWS ▾ PRESS RELEASES SPONSORED CONTENT EVENTS JOBS DIRECTORY RFP CENTER

Technology

Neural Conquers Patent Translation in Major WIPO Roll-out

by Marion Marking on November 18, 2016

Barely two years after it was first proposed, machine translation technology based on neural networks is going mainstream. After Google, Systran, and Microsoft, the World Intellectual Property Organization (WIPO) announced on October 31, 2016 the roll-out of neural machine translation (NMT) on its publicly

JOIN OUR MAILING L

Insightful language in
straight to your inbo

Email address

PUBLISHING PARTNE

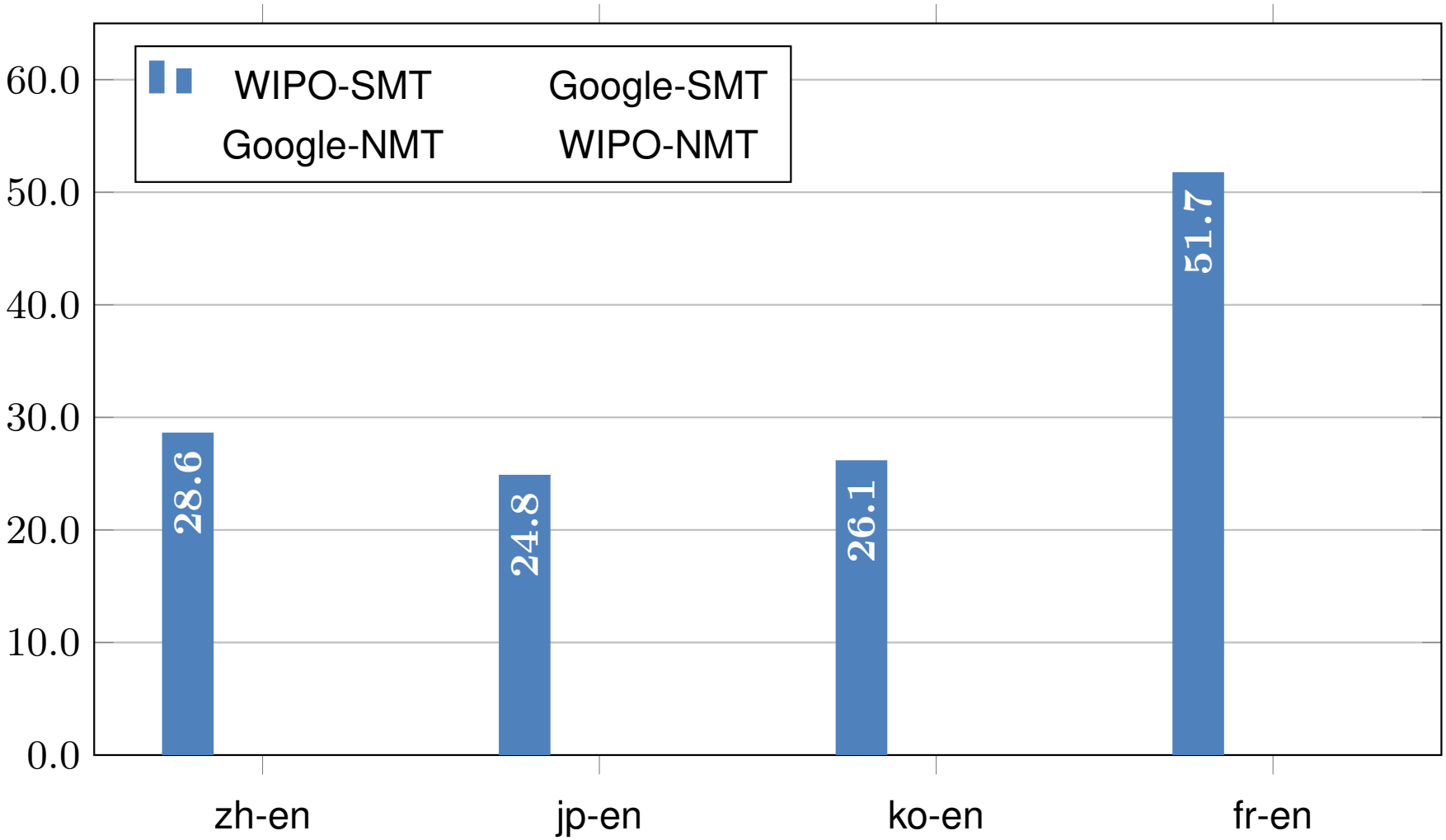
Break relevant news,
views, and bring your
your audience.

PUBLISH W

Bonus slide II

Deployment at WIPO (World Intellectual Property Organization)

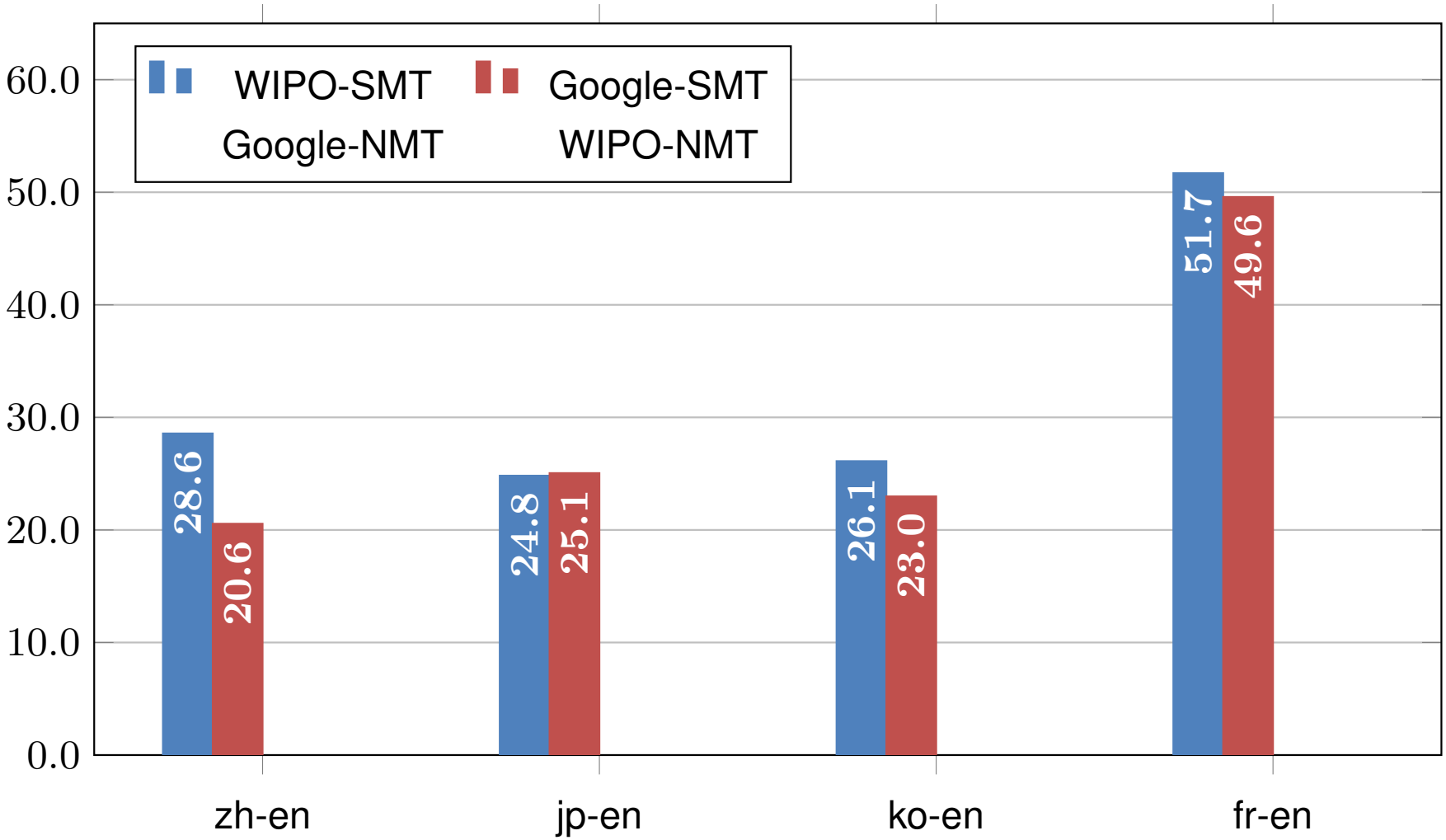
- Large corpora of in-domain patent data (30M-100M segments).



Bonus slide II

Deployment at WIPO (World Intellectual Property Organization)

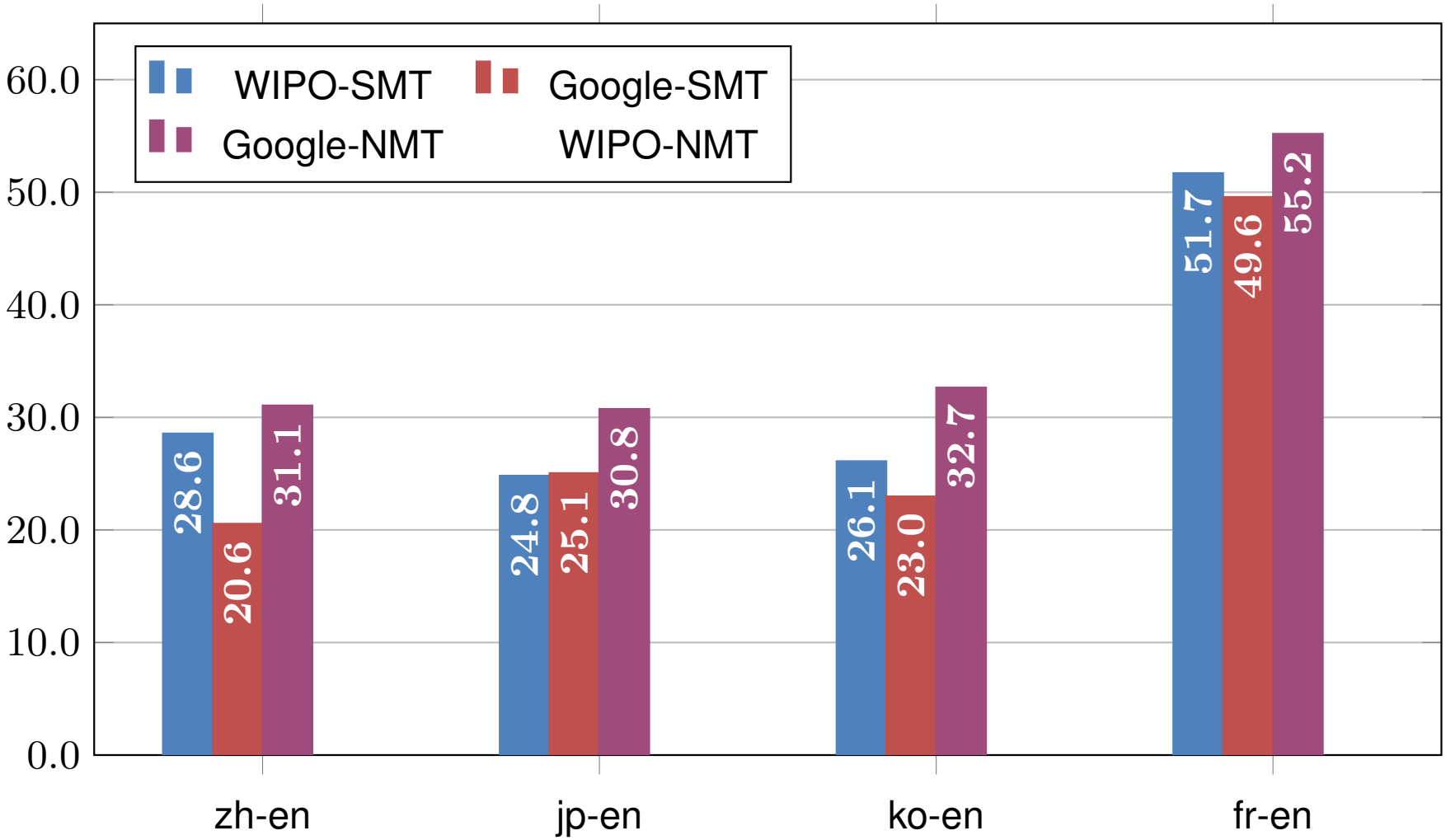
- Large corpora of in-domain patent data (30M-100M segments).



Bonus slide II

Deployment at WIPO (World Intellectual Property Organization)

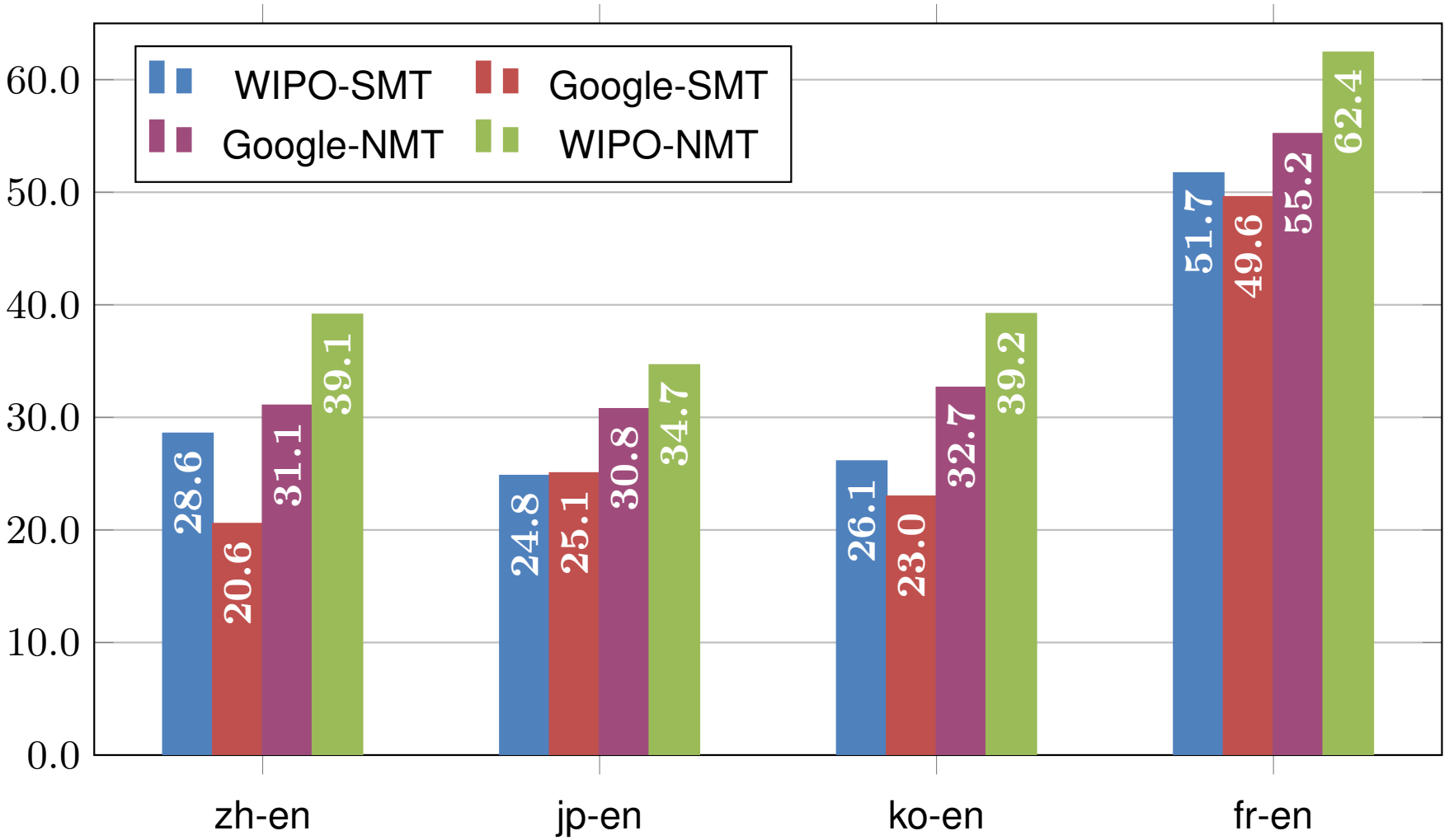
- Large corpora of in-domain patent data (30M-100M segments).



Bonus slide II

Deployment at WIPO (World Intellectual Property Organization)

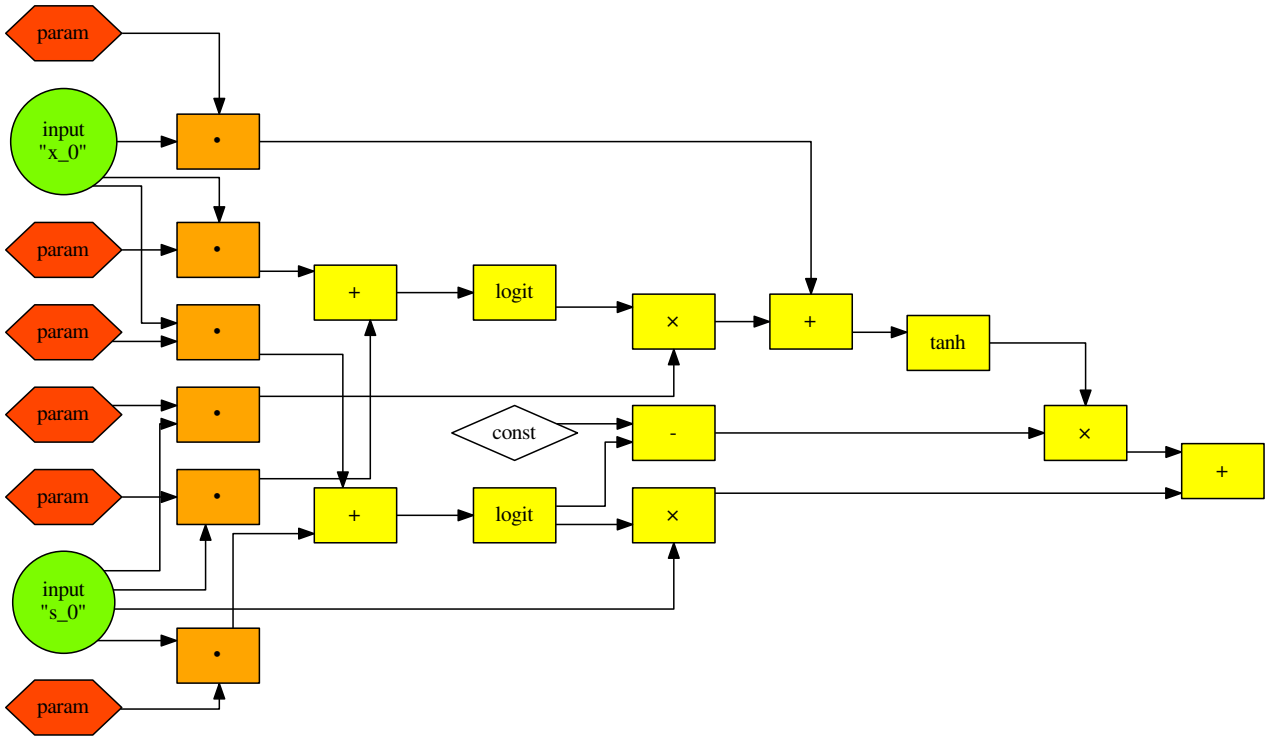
- Large corpora of in-domain patent data (30M-100M segments).



Bonus slide III

Custom C++ training framework

- Working on faster Nematus-compatible training framework;
- Toy models 2-3x faster than Theano or Tensorflow.



References


Bahdanau, D., Cho, K., and Bengio, Y. (2014).
Neural machine translation by jointly learning to align and translate.
CoRR, abs/1409.0473.


Jean, S., Cho, K., Memisevic, R., and Bengio, Y. (2015).
On using very large target vocabulary for neural machine translation.
In *ACL*, pages 1–10.


Johnson, H., Martin, J. D., Foster, G. F., and Kuhn, R. (2007).
Improving translation quality by discarding most of the phrasetable.
In *EMNLP-CoNLL*, pages 967–975.


Mi, H., Wang, Z., and Ittycheriah, A. (2016).
Vocabulary manipulation for neural machine translation.
In *ACL*.


Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013).
Efficient estimation of word representations in vector space.
CoRR, abs/1301.3781.