Neural Quality Estimation as a Bridge for Human-Computer Translation Symbiosis

Dimitar Shterionov

DUBLIN MACHINE TRANSLATION SUMMIT 2019

Neural Quality Estimation as a precondition for establishing a Bridge for Human-Computer Translation Symbiosis

Dimitar Shterionov

Based on joint work with:
Félix do Carmo, Joss Moorkens, Murhaf Hossari,
Eric Paquin, Dag Schmidtke, Declan Groves, Andy Way
Presented at: WMT2019 and MTSummit2019

Agenda

Overview of Quality Estimation and QE Systems

Automatic Post Editing

BERT(ology)

Our use-case

Conclusions, Future work, Ideas

Quality Estimation

- **Definition:**
  - Quality estimation (QE) (Specia et al., 2009) is the process of predicting the quality of a **machine translation (MT) system** without human intervention or reference translations.
  - QE can be at word-, sentence-, or document-level. In the case of document- and sentence-level, the task is typically to predict a score that corresponds to a target evaluation criteria or metric (typically HTER), i.e. it is a regression task.
- **Purpose:**
  - MT quality assessment
  - QE feedback into CAT Tools (Turchi et al. 2015, Specia, 2011)
  - Aid for post-editing: select/ignore, estimate time/effort (Juan Rowda 2016)
  - QE for automatically generated eCommerce browse pages titles (Ueffing et al 2018)

[Specia 2011] Exploiting Objective Annotations for MeasuringTranslation Post-editing Effort
[Turchi et al. 2015] MT Quality Estimation for Computer-assisted Translation:Does it Really Help?
[Ueffing et al. 2018] Quality Estimation for Automatically Generated Titles of eCommerce Browse Pages
[Rowda, 2016] A Language Approach to Machine Translation Quality Estimation
[Specia and Shah, 2018] Machine Translation Quality Estimation: Applications and Future Perspectives

Quality Estimation

- **Purpose:**
  - Aid for post-editing: select/ignore, estimate time/effort
    sentence-level

  - Improve efficiency and quality
  - Create further data for training QE and APE systems

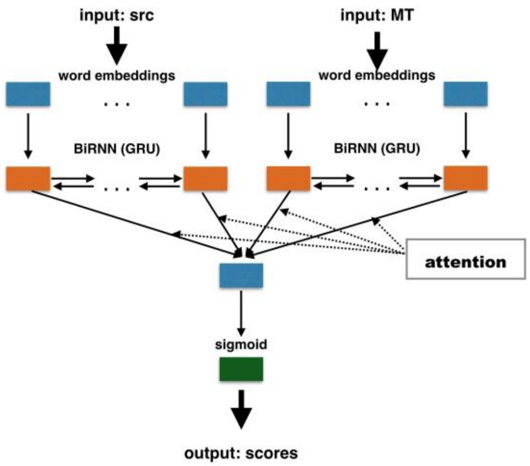
- **Word-level? Document-level?**

Neural QE systems

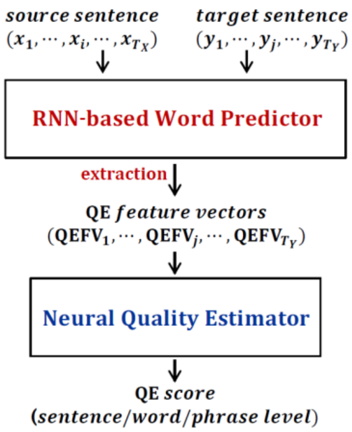
- WMT2018: DeepQuest; QEBrain; UNQE and MQE (non-neural system)
- MTSummit2019: SiameseQE
- WMT2019: UNBABEL (OpenKiwi), CMULTIMLT, NJUNLP BiQE, UTartu

Neural QE systems

- Single-phase:
  - From text to QE scores in one shot
  - No feature extraction

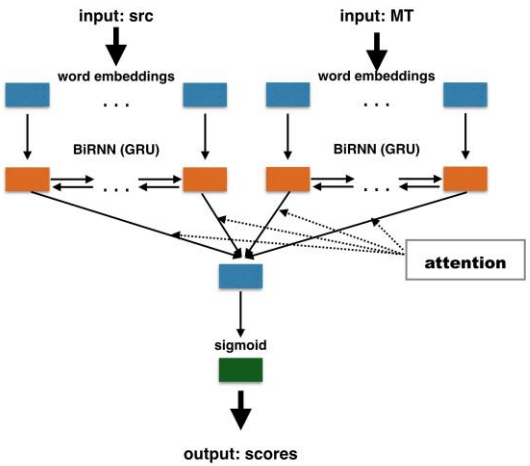


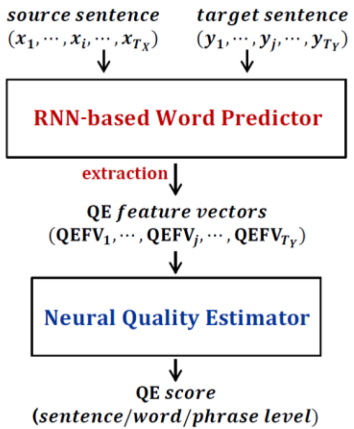
- Two-phase:
  - First phase trained on bilingual data to extract features
  - Second phase trained to compute QE scores from previously extracted features



Neural QE systems

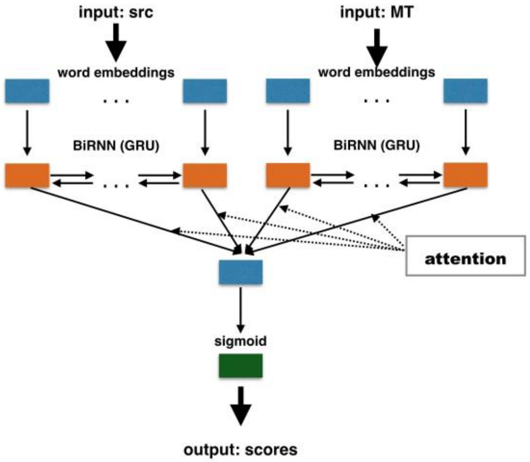
- Single-phase:
  - From text to QE scores in one shot
  - No feature extraction
- Two-phase:
  - First phase trained on bilingual data to extract features
  - Second phase trained to compute QE scores from previously extracted features
- BERT-phase:
  - Replace first phase with a pretrained embedding model for feature extraction
  - Second phase trained to compute QE scores from previously extracted features



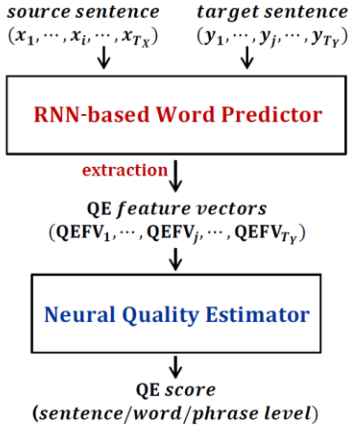


Neural QE systems

- Single-phase:
  - DeepQuest (BiRBNN)
  - SiameseQE



- Two-phase:
  - Postech (DeepQuest, OpenKiwi)
  - UNQE
  - NJUNLP (WMT2019)
  - USAAR-DFKI (WMT2019)



- BERT-phase:
  - OpenKiwi
  - MIPT (word-level only)
  - ETRI
  - CMU
  - UTartu

Use-case: software translation

- QE of the translations of software UI strings from Microsoft products
  - Domain: Technical/IT
  - Proprietary and open-access data (EN-DE and EN-ES)
  - HTER
- Research questions:
  - Can Neural Network approaches to Quality Estimation (QE) help increase the identification of publishable machine translation (MT) content?
  - Can the new approaches be easily implemented in a corporate setting?
- Evaluation:
  - Business metrics
  - Performance metrics
  - Cost

Use-case. Data

- Domain: Technical/IT
- Proprietary and open-access data (EN-DE and EN-ES)
- HTER

QE data	EN-DE	EN-ES
Train	67 718	46 217
Dev	7 524	5 136
Test	32 898	34 623

Table 1. Number of sentence pairs used for QE training and evaluation.

Extra data	EN-DE	EN-ES
Europarl	1 863 144	1 850 469
Microsoft	1 741 218	1 581 875

Table 2. Number of parallel sentences used for first-phase training.

System setup: hardware

- GPU-powered machines:
    - First:
        - 2 x nVidia TitanX,
        - 64 GB RAM
        - Intel(R) Core(TM) i7-5960X CPU
    - Second:
        - 4 x nVidia GTX 1080Ti
        - 128 GB RAM
        - Intel(R) Core(TM) i7-7820X CPU.
- Each models trained and evaluated using one GPU
    - QEBrain used 4 GPUs in parallel to train due to computational power required

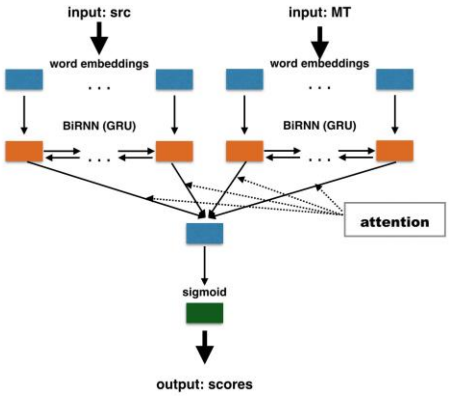
System setup: software

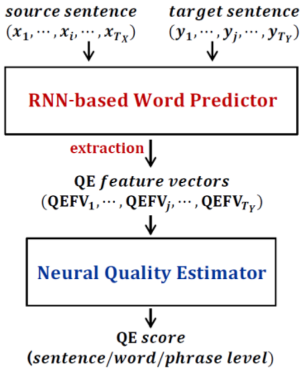
- Three different Anaconda 3 virtual environments:
  - For **deepQuest**: Python v2.7, theano, keras, numpy, image v1.5.27 and scikit-learn.
  - For **QEBrain**: Python v3.6.6, tensorflow-gpu v1.12,0, opennmt-tf v1.15.0, numpy, scipy and scikit-learn.
  - For **SiameseQE**: Python v3.6.6, pytorch v0.3.1, numpy

- Evaluation script:
  - a script that computed the scores from all three toolkits in the same way.
  - numpy and scikit-learn to compute Pearson coreference coefficient (pearson), Rooted Mean Squared Error (RMSE) and Mean Absolute Error (MAE).

Trained systems - DeepQuest

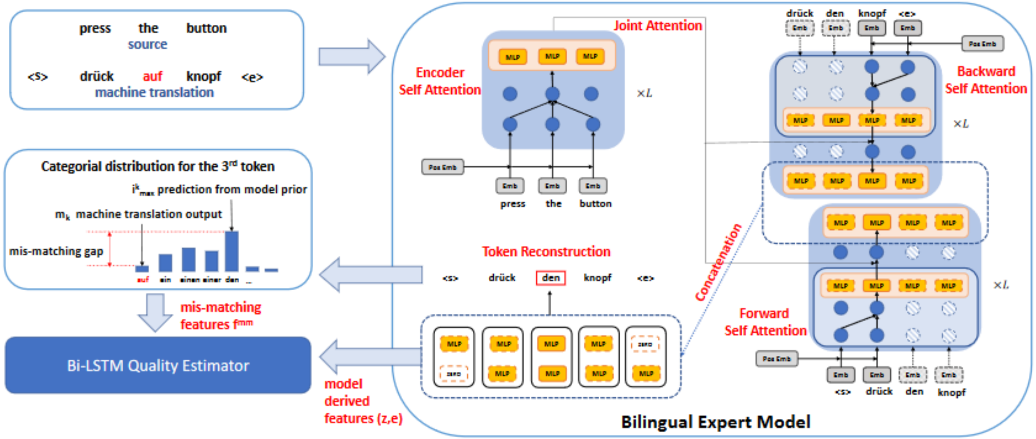
- 2 Methods (single-phase and a two-phase):
  - BiRNN – two GRU RNNs with attention
  - POSTECH – a predictor and an estimator
- Implementation:
  - theano + keras
  - Actively developed by Sheffield
- Complexity:
  - Implementation: medium to high
  - Execution: high





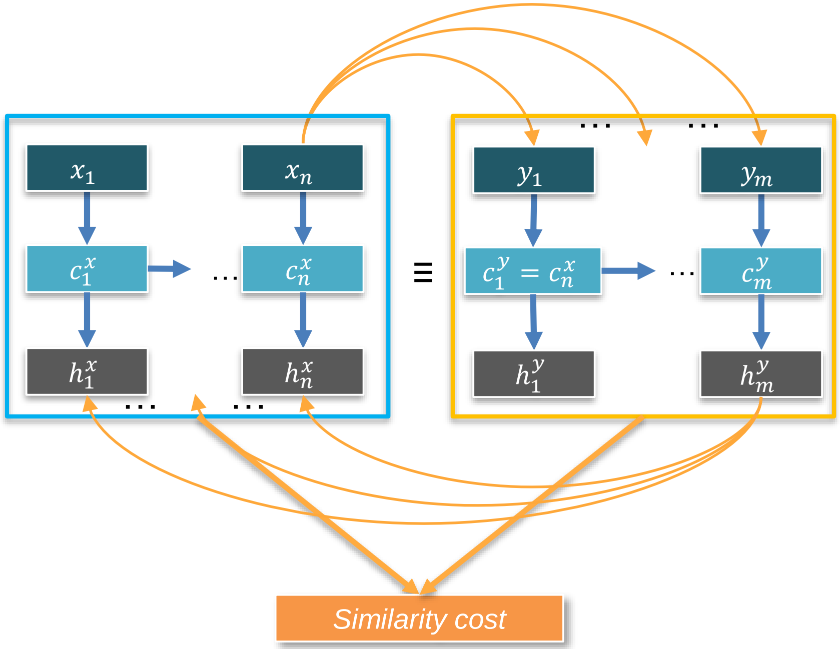
Trained systems - QEBrain

- Only one method (two-phase)
  - Expert model
  - QE model
- Implementation:
  - Tensorflow
  - First public release
  - Buggy, hardcoded
- Complexity:
  - Implementation: high
  - Execution: low



Trained systems - SiameseQE

- 3 methods (single-phase)
  - NoATT
  - DotATT
  - w2wATT
- Implementation
  - Pytorch (v0.3.1)
  - 2-layer, bidirectional LSTM
- Complexity:
  - Implementation: low-medium
  - Execution: low-medium



Evaluation - business metrics

EN-DE	AUC
QEBrain	**0.8091**
BiRNN	**0.7475**
Siam DotATT	**0.7342**
Postech EU	0.7154
Postech MSFT	0.7047
Siam w2wAtt	0.6698
33features	0.6639
Siam NoATT	0.6004

EN-DE	Throughput
QEBrain	**13.35%**
BiRNN	**12.63%**
Siam DotATT	**12.57%**
Siam w2wAtt	12.43%
Postech EU	12.38%
Postech MSFT	11.95%
33features	11.10%
Siam NoATT	10.39%

EN-DE	Gain
QEBrain	**3.55%**
BiRNN	**2.83%**
Siam DotATT	**2.77%**
Siam w2wAtt	2.63%
Postech EU	2.58%
Postech MSFT	2.15%
33features	1.30%
Siam NoATT	0.59%

EN-DE	Precision
QEBrain	**40.33%**
Siam DotATT	**37.39%**
BiRNN	**36.97%**
Postech EU	36.74%
Siam w2wAtt	35.67%
Postech MSFT	34.50%
33features	29.24%
Siam NoATT	26.64%

EN-DE	Distance to ideal
QEBrain	**2.14%**
BiRNN	**2.86%**
Siam DotATT	**2.92%**
Siam w2wAtt	3.06%
Postech EU	3.11%
Postech MSFT	3.54%
33features	4.39%
Siam NoATT	5.10%

EN-ES	AUC
QEBrain	**0.7259**
Postech MSFT	**0.6708**
BiRNN	**0.6683**
33features	0.6617
Siam DotATT	0.6557
Postech EU	0.6401
Siam w2wAtt	0.6008
Siam NoATT	0.5359

EN-ES	Throughput
QEBrain	**22.82%**
Postech MSFT	**21.92%**
Siam DotATT	**21.87%**
BiRNN	21.77%
33features	21.63%
Siam w2wAtt	21.36%
Postech EU	21.01%
Siam NoATT	16.65%

EN-ES	Gain
QEBrain	**6.06%**
Postech MSFT	**5.16%**
Siam DotATT	**5.12%**
BiRNN	5.02%
33features	4.88%
Siam w2wAtt	4.60%
Postech EU	4.26%
Siam NoATT	-0.11%

EN-ES	Precision
QEBrain	**65.38%**
Siam DotATT	**63.62%**
Postech MSFT	**63.61%**
BiRNN	63.42%
33features	63.14%
Siam w2wAtt	62.71%
Postech EU	62.10%
Siam NoATT	54.95%

EN-ES	Distance to ideal
QEBrain	**6.50%**
Postech MSFT	**7.40%**
Siam DotATT	**7.45%**
BiRNN	7.55%
33features	7.69%
Siam w2wAtt	7.96%
Postech EU	8.31%
Siam NoATT	12.67%

Evaluation - model performance

EN-DE	Pearson's *r* ↑
QEBrain	**0.62321**
BiRNN	**0.48107**
33features	**0.45845**
Siam DotATT	**0.42774**
Postech MSFT	0.42546
Postech EU	0.41017
Siam w2wATT	0.28689
Siam NoATT	0.25351

EN-DE	MAE ↓
QEBrain	**0.17534**
BiRNN	**0.21068**
33features	**0.21242**
Siam DotATT	**0.21321**
Postech MSFT	0.21534
Postech EU	0.21940
Siam w2wATT	0.25453
Siam NoATT	0.25547

EN-DE	RMSE ↓
QEBrain	**0.24160**
33features	**0.27292**
Siam DotATT	**0.27545**
Postech MSFT	0.27697
BiRNN	0.28190
Postech EU	0.28378
Siam NoATT	0.31760
Siam w2wATT	0.36092

EN-ES	Pearson's *r* ↑
QEBrain	**0.52354**
33features	**0.36504**
Postech MSFT	**0.36360**
BiRNN	0.35991
Siam DotATT	**0.32057**
Postech EU	0.30554
Siam w2wATT	0.29931
Siam NoATT	0.11151

EN-ES	MAE ↓
QEBrain	**0.18564**
Siam NoATT	0.22162
BiRNN	**0.22263**
Postech MSFT	0.22918
Siam DotATT	**0.22971**
33features	0.23493
Postech EU	0.25340
Siam w2wATT	0.30841

EN-ES	RMSE ↓
QEBrain	**0.24546**
Siam NoATT	0.27497
Siam DotATT	**0.28984**
BiRNN	0.29139
33features	0.29354
Postech MSFT	0.29748
Postech EU	0.32144
Siam w2wATT	0.42366

Evaluation - model performance

EN-DE	Pearson's *r* ↑
QEBrain	**0.62321**
BiRNN	**0.48107**
33features	**0.45845**
Siam DotATT	**0.42774**
Postech MSFT	0.42546
Postech EU	0.41017
Siam w2wATT	0.28689
Siam NoATT	0.25351

EN-DE	MAE ↓
QEBrain	**0.17534**
BiRNN	**0.21068**
33features	**0.21242**
Siam DotATT	**0.21321**
Postech MSFT	0.21534
Postech EU	0.21940
Siam w2wATT	0.25453
Siam NoATT	0.25547

EN-DE	RMSE ↓
QEBrain	**0.24160**
33features	**0.27292**
Siam DotATT	**0.27545**
Postech MSFT	0.27697
BiRNN	0.28190
Postech EU	0.28378
Siam NoATT	0.31760
Siam w2wATT	0.36092

EN-DE	Adjusted ranking
QEBrain	**0.67264**
BiRNN	**0.31692**
33features	**0.29380**
Siam DotATT	0.24161
Postech MSFT	0.23115
Postech EU	0.18827
Siam NoATT	-0.18029
Siam w2wAtt	-0.19898

EN-ES	Pearson's *r* ↑
QEBrain	**0.52354**
33features	**0.36504**
Postech MSFT	**0.36360**
BiRNN	0.35991
Siam DotATT	**0.32057**
Postech EU	0.30554
Siam w2wATT	0.29931
Siam NoATT	0.11151

EN-ES	MAE ↓
QEBrain	**0.18564**
Siam NoATT	0.22162
BiRNN	**0.22263**
Postech MSFT	0.22918
Siam DotATT	**0.22971**
33features	0.23493
Postech EU	0.25340
Siam w2wATT	0.30841

EN-ES	RMSE ↓
QEBrain	**0.24546**
Siam NoATT	0.27497
Siam DotATT	**0.28984**
BiRNN	0.29139
33features	0.29354
Postech MSFT	0.29748
Postech EU	0.32144
Siam w2wATT	0.42366

EN-ES	Adjusted ranking
QEBrain	**0.49396**
BiRNN	**0.09304**
Postech MSFT	**0.07473**
33features	0.07117
Siam DotATT	0.02116
Postech EU	-0.10364
Siam NoATT	-0.25295
Siam w2wAtt	-0.39747

$$\omega_i = (0.5 + \frac{0.5 \times r_i}{\overline{r}}) - (\frac{MAE_i}{\overline{MAE}} + \frac{RMSE_i}{\overline{RMSE}})/2$$

Comparison remarks

- QEBrain – the best performing system.
- Statistical/feature-based system:
  - Among the better ranking systems
  - Optimised to fit business model
- Not all two-phase systems better than one-phase systems
- How about…

Comparison remarks

- QEBrain – the best performing system.
- Statistical/feature-based system:
  - Among the better ranking systems
  - Optimised to fit business model
- Not all two-phase systems better than one-phase systems

- How about… cost?

$$\$€£ = \alpha * (t + CO_2)$$

Consumed resources

- Time

System	GPU	Original time (minutes)					
		EN-DE			EN-ES		
		I	II	**Total**	I	II	**Total**
BiRNN	T	–	–	**265**	–	–	**152**
Post. EU	T	1 770	262	**2 032**	1 859	159	**2 018**
Post. MS	T	1 118	160	**1 268**	1 752	154	**1 906**
QEBrain	G	859	107	**966**	863	91	**954**
S. NoATT	G	–	–	**37**	–	–	**86**
S. DotATT	G	–	–	**102**	–	–	**80**
S. w2wATT	G	–	–	**75**	–	–	**62**

Consumed resources

- Time

System	GPU	Original time (minutes)						Adjusted time (minutes). GPU Speed coef. = 0.45					
		EN-DE			EN-ES			EN-DE			EN-ES		
		I	II	Total	I	II	Total	I	II	Total	I	II	Total
BiRNN	T	–	–	**265**	–	–	**152**	–	–	**119**	–	–	**68**
Post. EU	T	1 770	262	**2 032**	1 859	159	**2 018**	797	118	**915**	837	72	**908**
Post. MS	T	1 118	160	**1 268**	1 752	154	**1 906**	503	72	**575**	788	69	**858**
QEBrain	G	859	107	**966**	863	91	**954**	3 436	107	**3 543**	3 452	91	**3 543**
S. NoATT	G	–	–	**37**	–	–	**86**	–	–	**37**	–	–	**86**
S. DotATT	G	–	–	**102**	–	–	**80**	–	–	**102**	–	–	**80**
S. w2wATT	G	–	–	**75**	–	–	**62**	–	–	**75**	–	–	**62**

Consumed resources

- Time

System	GPU	Original time (minutes)						Adjusted time (minutes). GPU Speed coef. = 0.45					
		EN-DE			EN-ES			EN-DE			EN-ES		
		I	II	**Total**	I	II	**Total**	I	II	**Total**	I	II	**Total**
BiRNN	T	–	–	**265**	–	–	**152**	–	–	**119**	–	–	**68**
Post. EU	T	1 770	262	**2 032**	1 859	159	**2 018**	797	118	**915**	837	72	**908**
Post. MS	T	1 118	160	**1 268**	1 752	154	**1 906**	503	72	**575**	788	69	**858**
QEBrain	G	859	107	**966**	863	91	**954**	3 436	107	**3 543**	3 452	91	**3 543**
S. NoATT	G	–	–	**37**	–	–	**86**	–	–	**37**	–	–	**86**
S. DotATT	G	–	–	**102**	–	–	**80**	–	–	**102**	–	–	**80**
S. w2wATT	G	–	–	**75**	–	–	**62**	–	–	**75**	–	–	**62**

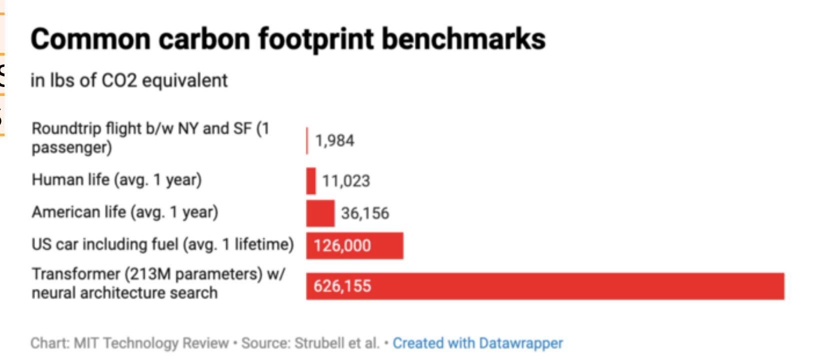
- GPU Memory:

System	Memory (%)
BiRNN	70-90
Postech	85-100
QEBrain	98-100
Siamese	60-80

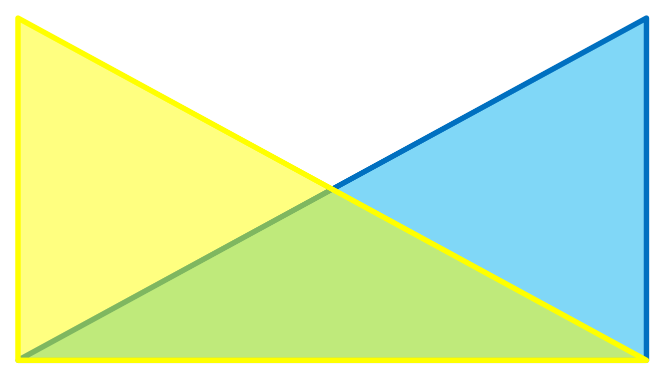
Consumed resources

- Time

System	GPU	Original time (minutes)						Adjusted time (minutes). GPU Speed coef. = 0.45					
		EN-DE			EN-ES			EN-DE			EN-ES		
		I	II	**Total**	I	II	**Total**	I	II	**Total**	I	II	**Total**
BiRNN	T	–	–	**265**	–	–	**152**	–	–	**119**	–	–	**68**
Post. EU	T	1 770	262	**2 032**	1 859	159	**2 018**	797	118	**915**	837	72	**908**
Post. MS	T	1 118	160	**1 268**	1 752	154	**1 906**	503	72	**575**	788	69	**858**
QEBrain	G	859	107	**966**	863	91	**954**						**543**
						–	**86**						**86**
S						–	**80**						**80**
S						–	**62**						**62**



	Memory (%)
N	70-90
ch	85-100



[Strubell et al. 2019] Energy and Policy Considerations for Deep Learning in NLP
[Lukas Biewald 2019] Deep Learning and Carbon Emissions

Machine Translation

Quality Translation Output

Automatic Post-Editing with a context token

- **WMT2019 Shared task submission:**

APE through neural and statistical MT with augmented data
ADAPT/DCU submission to the WMT 2019 APE Shared task

- **Motivation**
  - Extra context (specific global properties of groups of segments) added as a prefix or a suffix to each segment
  - Successful in domain adaptation of MT and APE, this technique deserves further attention
  - Prefixes per:
    - Meaning: **Topic models**
    - Structure: **Sentence length**

Automatic Post-Editing with a context token

- **(Neural) approach**
  - multi-source systems trained on context-augmented data
  - Marian-NMT (multi-s2s) with LSTM units
  - Early stopping after 5 epochs
- **Bins (for NPE):**
  - **Topic models**
    - Latent Dirichlet Allocation (LDA)
    - On the source (English) side
    - Ten topic clusters
  - **Sentence length**
    - # of tokens in the source sentence
    - 8 bins of similar sizes
- **Input augmentation:**

<TOPIC1> In addition , four-color gra@@ ys using different hues are included .
<TOPIC1> Darüber hinaus werden vier Grautöne mit unterschiedlichen Graut@@ önen verwendet .

Automatic Post-Editing with a context token

- **Data**
  - Authentic and synthetic data
  - Divided into 3 datasets for EN-DE and 2 datasets for EN-RU

Size	EN-DE	EN-RU
small	268 840	301 780
medium	795 208	N/A
large	4 660 020	8 037 141

Table 1: Number of SRC-NMT-PE triplets distributed over three data sets.

Size	EN-DE			EN-RU		
	SRC	NMT	PE	SRC	NMT	PE
small	10 771	15 477	18 088	9 125	14 783	15 761
medium	48 227	48 257	48 869	N/A		
large	50 327	50 538	50 790	53 030	50 646	52 970

Table 2: Vocabulary sizes (after applying BPE).

- **Results**
  - **EN-DE:**

	Model	Prefix	BLEU ↑	TER ↓
MT	Baseline	N/A	76.94	15.08
NPE	small	N/A	63.28	24.09
	medium	N/A	70.57	18.81
	large	N/A	70.29	19.89
	small	topic	60.41	28.59
	medium	topic	73.08	17.81
	large	**topic**	**75.82**	**15.89**
	small	length	62.56	26.91
	medium	length	73.74	17.26
	large	**length**	**75.85**	**15.91**

  - **EN-RU:**

	Model	Prefix	BLEU ↑	TER ↓
MT	Baseline	N/A	80.22	13.13
NPE	small	N/A	50.76	34.45
	large	N/A	59.01	28.01
	small	topic	48.30	41.19
	large	topic	75.39	16.18
	small	length	44.68	44.57
	large	**length**	**73.67**	**19.74**

Automatic Post-Editing with a context token

- **(Neural) approach**
  - multi-source systems trained on context-augmented data
  - Marian-NMT (multi-s2s) with LSTM units
  - Early stopping after 5 epochs
- **Bins (for NPE):**
  - **Topic models**
    - Latent Dirichlet Allocation (LDA)
    - On the source (English) side
    - Ten topic clusters
  - **Sentence length**
    - # of tokens in the source sentence
    - 8 bins of similar sizes
  - **Quality information**
    - (H)TER scores
    - 4 bins
    - Unavailable for test set
- **Input augmentation:**

<TOPIC1> In addition , four-color gra@@ ys using different hues are included .

<TOPIC1> Darüber hinaus werden vier Grautöne mit unterschiedlichen Graut@@ önen verwendet .

Automatic Post-Editing with a TER token

- **QE / TER Bins:**
  - **4 bins: 0.0, 0.0-0.3, 0.3-0.7, 0.7-1.0**
  - **Extracted from APE training data: MT<->PE**
  - **Used to train QE systems**
- **Results:**

	BLEU	TER
Gold standard	73.04	17.1
Trained Emb	72.3	17.9
BERT	72	18
None tag	67.6	23.6
Random tag	68	21.2

Automatic Post-Editing with a TER token

- **QE / TER Bins:**
  - **4 bins: 0.0, 0.0-0.3, 0.3-0.7, 0.7-1.0**
  - **Extracted from APE training data: MT<->PE**
  - **Used to train QE systems**
- **Results:**

	BLEU	TER
Gold standard	73.04	17.1
Trained Emb	72.3	17.9
BERT	72	18
None tag	67.6	23.6
Random tag	68	21.2

- **Next steps:**
  - **Classification and not regression**
  - **Intelligent division**
  - **Human Post-Editing**

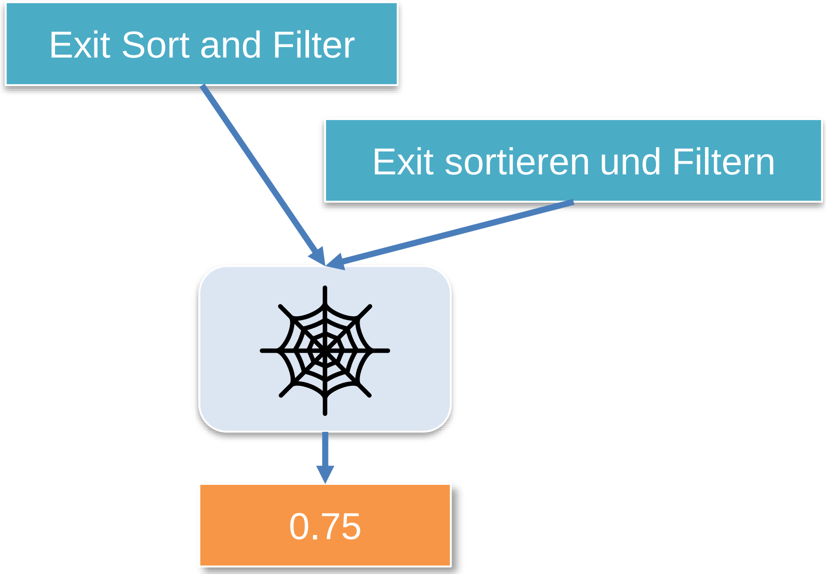
BERT(ology)

- ELMo
  - Contextual: The representation for each word depends on the entire context in which it is used.
  - Deep: The word representations combine all layers of a deep pre-trained neural network.
  - Two-layer biLMs with character convolutions.
  - Character based: allows robust representations for out-of-vocabulary tokens unseen in training.
- BERT
  - Uses masked language models to enable pre-trained deep bidirectional representations (Transformer).
  - Next sentence prediction
  - A multi-layer bidirectional Transformer encoder.
  - Mono- and multi-lingual pretrained models.
  - Tough BERT was trained on over 100 languages, it wasn't optimized for multi-lingual models — most of the vocabulary isn't shared between languages and therefore the shared knowledge is limited.
- XLM
  - Cross-lingual language model with shared BPE vocabulary for improved alignment of embedding spaces across languages.
  - Causal language model, Masked language model (as in BERT) – monolingual – and Translation language model – parallel data.
  - Each training sample consists of the same text in two languages.
  - The model also receives the language ID and the order of the tokens in each language, i.e. the Positional Encoding, separately which helps the model learn the relationship between related tokens in different languages.

Vector spaces

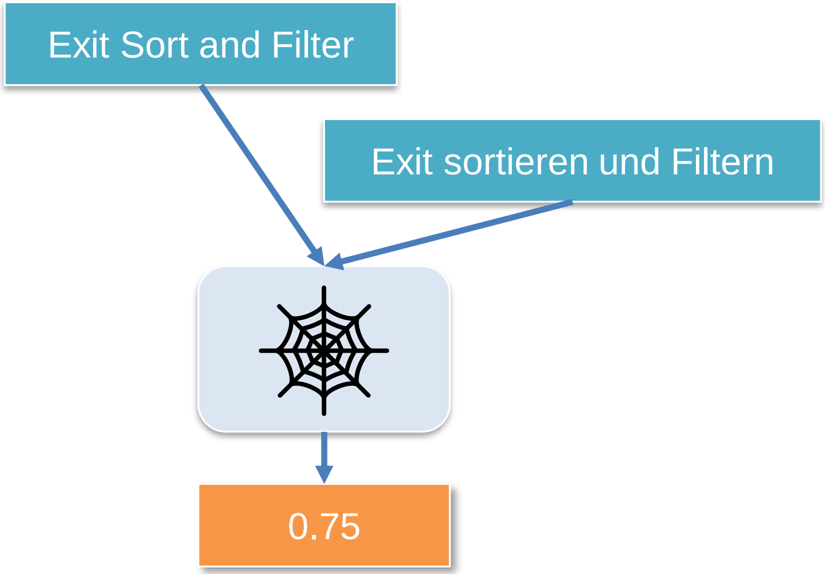
1. Convert sentences into vector representations in some vector space

2. Compute distance/similarity between vectors



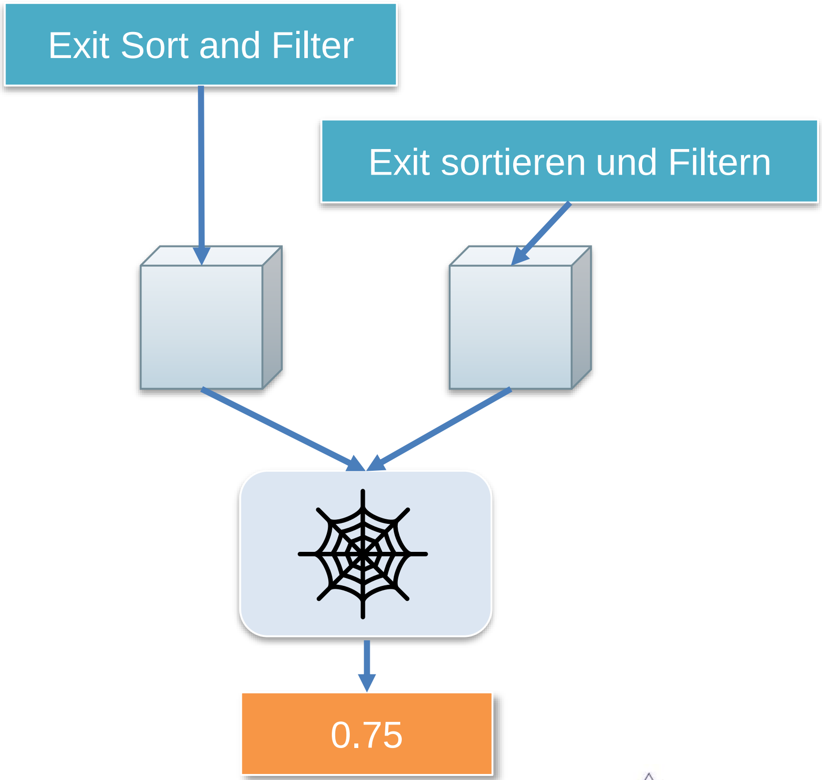
Vector spaces

1. Convert sentences into vector representations in some vector space
    - English and German vocabularies can be similar => English and German different vector spaces can be actually similar
    - English and Bulgarian vocabularies are quite different => different vector spaces
2. Compute distance/similarity between vectors
    - The network could learn how to interpret distance/similarity as (H)TER. Easier
    - The network could also learn how to reduce the differences between the vector spaces. More complex.



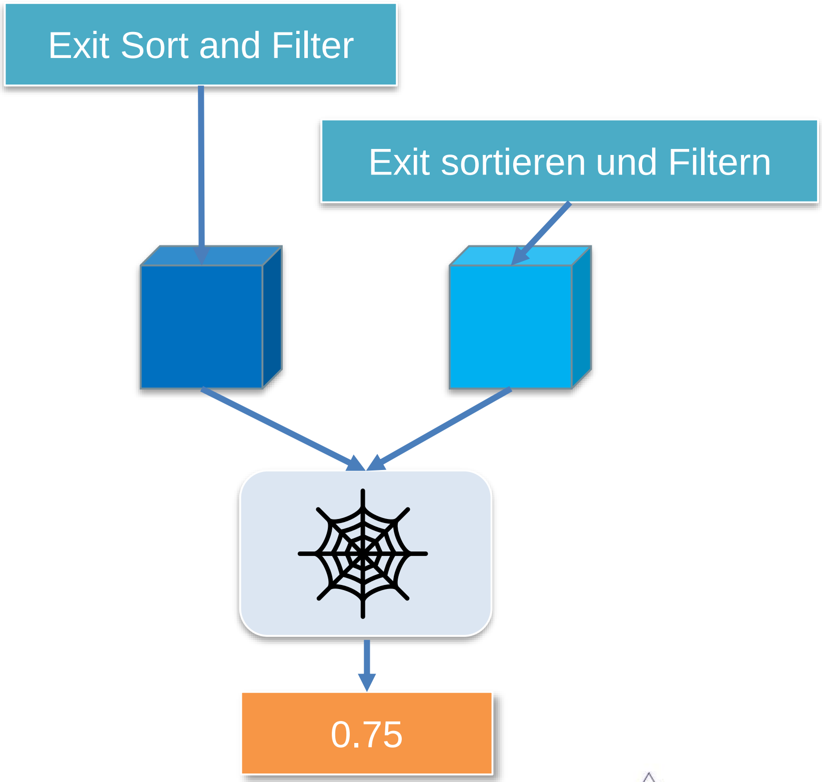
Vector spaces

1. Convert sentences into vector representations in some vector space
   - English and German vocabularies can be similar => English and German different vector spaces can be actually similar
   - English and Bulgarian vocabularies are quite different => different vector spaces
2. Compute distance/similarity between vectors
   - The network could learn how to interpret distance/similarity as (H)TER. Easier
   - The network could also learn how to reduce the differences between the vector spaces. More complex.



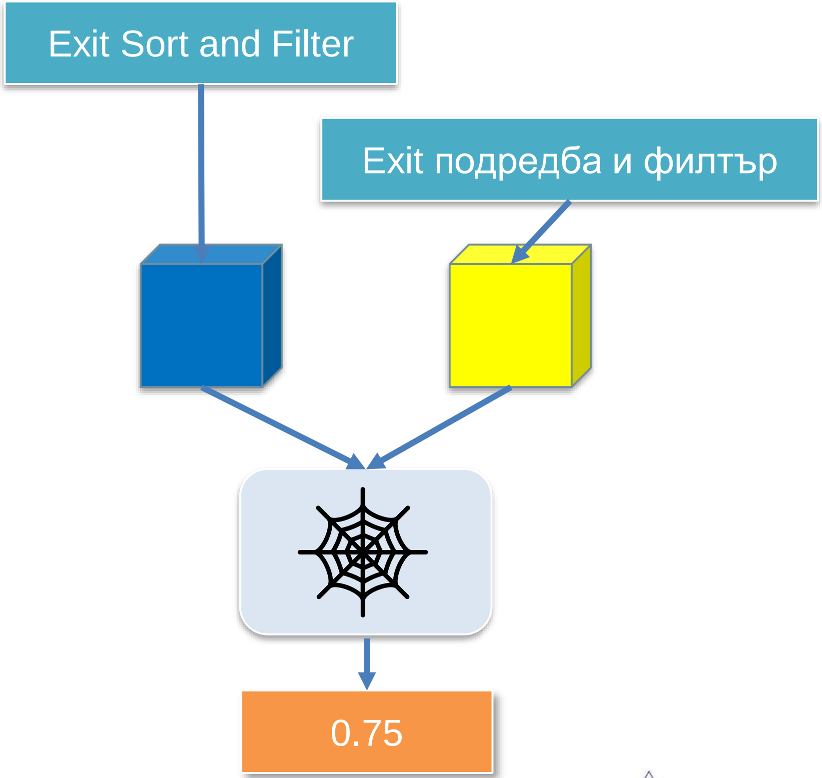
Vector spaces

1. Convert sentences into vector representations in some vector space
   - English and German vocabularies can be similar => English and German different vector spaces can be actually similar
   - English and Bulgarian vocabularies are quite different => different vector spaces
2. Compute distance/similarity between vectors
   - The network could learn how to interpret distance/similarity as (H)TER. Easier
   - The network could also learn how to reduce the differences between the vector spaces. More complex.



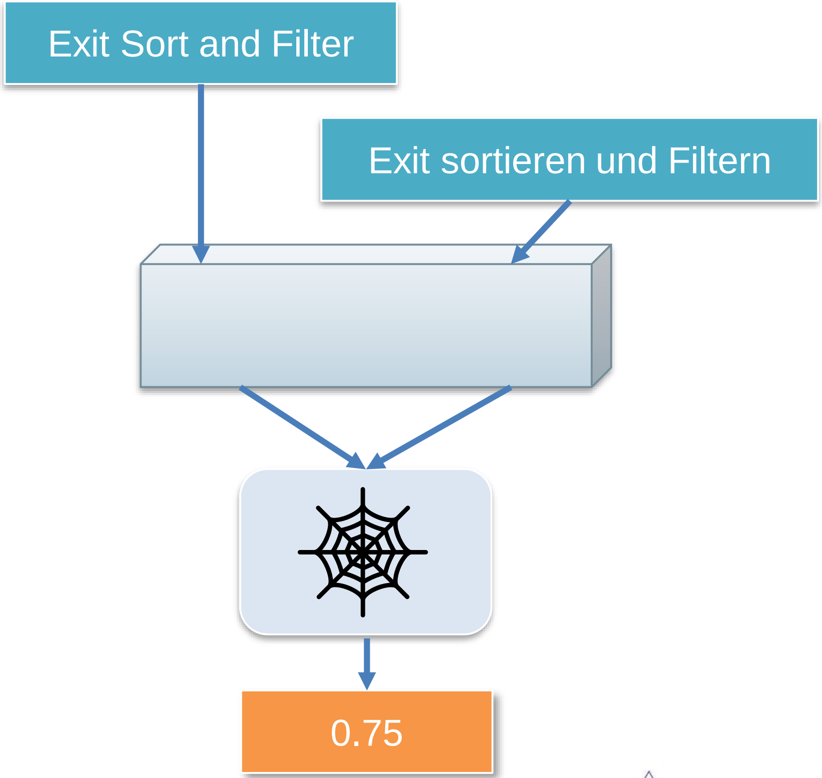
Vector spaces

1. Convert sentences into vector representations in some vector space
    - English and German vocabularies can be similar => English and German different vector spaces can be actually similar
    - English and Bulgarian vocabularies are quite different => different vector spaces
2. Compute distance/similarity between vectors
    - The network could learn how to interpret distance/similarity as (H)TER. Easier
    - The network could also learn how to reduce the differences between the vector spaces. More complex.



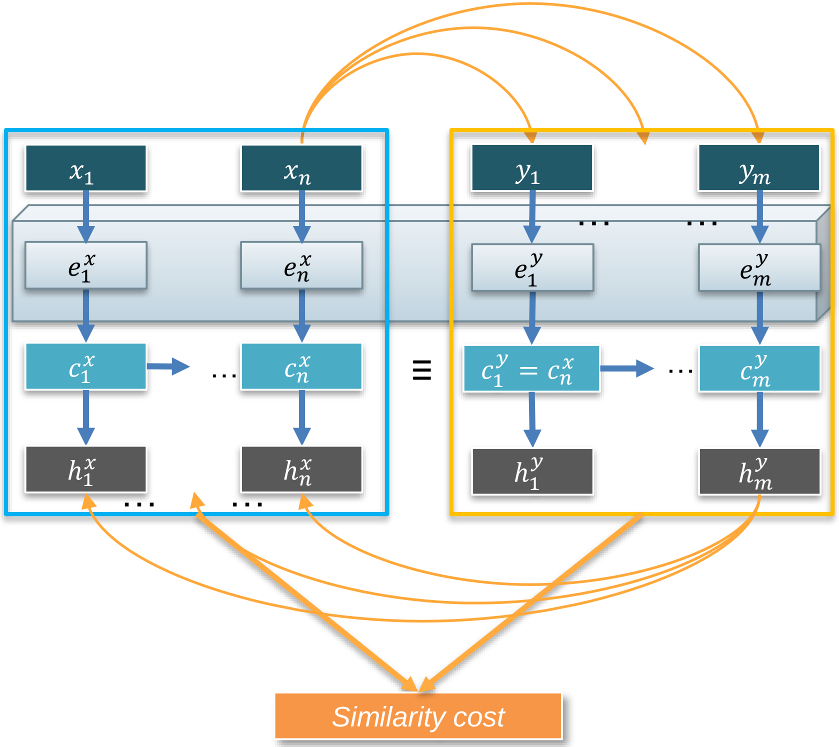
Vector spaces

1. Convert sentences into vector representations in some vector space
   - English and German vocabularies can be similar => English and German different vector spaces can be actually similar
   - English and Bulgarian vocabularies are quite different => different vector spaces
2. Compute distance/similarity between vectors
   - The network could learn how to interpret distance/similarity as (H)TER. Easier
   - The network could also learn how to reduce the differences between the vector spaces. More complex.



SiameseQE with pretrained embeddings

- Sentence representations originate from
  - BERT
  - XLM
- Embeddings are not learnable – no fine tuning of BERT/XLM
  - Huge models
  - Intermediate results can be cached



SiameseQE with pretrained embeddings

- Sentence representations originate from
  - BERT
  - XLM
- Embeddings are not learnable – no fine tuning of BERT/XLM
  - Huge models
  - Intermediate results can be cached
  - But results are not as expected ☹

System	EN-DE			EN-ES		
	Pearson (higher better)	MAE (lower better)	RMSE (lower better)	Pearson (higher better)	MAE (lower better)	RMSE (lower better)
BiRNN	0.48107	0.21068	0.2819	0.35991	0.22263	0.29139
POST. EU	0.41017	0.2194	0.28378	0.30554	0.2534	0.32144
POST. MSFT	0.42546	0.21534	0.27697	0.3636	0.22918	0.29748
QEBrain MSFT	0.62321	0.17534	0.2416	0.52354	0.18564	0.24546
Siamese NoATTN	0.25351	0.25547	0.3176	0.11151	0.22162	0.27497
Siamese DotATTN	0.42774	0.21321	0.27545	0.32057	0.22971	0.28984
Siamese w2wATTN	0.28689	0.25453	0.36092	0.29931	0.30841	0.42366
33features	0.45845	0.21242	0.27292	0.36504	0.23493	0.29354
Pretrained embeddings						
BERT BiRNN	0.4498	0.21108	0.27001	0.31244	0.21623	0.26914
BERT	0.42864	0.2134	0.27155	0.2609	0.24572	0.30859
XLM TLM	0.1864	0.24296	0.30153	0.10022	0.24055	0.2913
XLM	0.20977	0.2415	0.29796	0.10203	0.23377	0.28303

SiameseQE with pretrained embeddings

- Sentence representations originate from
  - BERT
  - XLM
- Embeddings are not learnable – no fine tuning of BERT/XLM
  - Huge models
  - Intermediate results can be cached
  - But results are not as expected ☹
- Need to fine-tune?

System	EN-DE			EN-ES		
	Pearson (higher better)	MAE (lower better)	RMSE (lower better)	Pearson (higher better)	MAE (lower better)	RMSE (lower better)
BiRNN	0.48107	0.21068	0.2819	0.35991	0.22263	0.29139
POST. EU	0.41017	0.2194	0.28378	0.30554	0.2534	0.32144
POST. MSFT	0.42546	0.21534	0.27697	0.3636	0.22918	0.29748
QEBrain MSFT	0.62321	0.17534	0.2416	0.52354	0.18564	0.24546
Siamese NoATTN	0.25351	0.25547	0.3176	0.11151	0.22162	0.27497
Siamese DotATTN	0.42774	0.21321	0.27545	0.32057	0.22971	0.28984
Siamese w2wATTN	0.28689	0.25453	0.36092	0.29931	0.30841	0.42366
33features	0.45845	0.21242	0.27292	0.36504	0.23493	0.29354
Pretrained embeddings						
BERT BiRNN	0.4498	0.21108	0.27001	0.31244	0.21623	0.26914
BERT	0.42864	0.2134	0.27155	0.2609	0.24572	0.30859
XLM TLM	0.1864	0.24296	0.30153	0.10022	0.24055	0.2913
XLM	0.20977	0.2415	0.29796	0.10203	0.23377	0.28303

SiameseQE with pretrained embeddings

- Sentence representations originate from
  - BERT
  - XLM
- Embeddings are not learnable – no fine tuning of BERT/XLM
  - Huge models
  - Intermediate results can be cached
  - But results are not as expected ☹
- Need to fine-tune?

System	Time per epoch (m)			
	EN-DE		EN-ES	
	Min	Max	Min	Max
BiRNN BERT	22.00	22.15	14.70	14.98
BERT	21.01	22.68	14.11	15.29
BERT-cache	1.05	19.85	0.66	13.77
XLM TLM	18.86	19.79	13.63	13.89
XLM	18.54	20.27	13.13	13.67

Conclusions & future work

- Advances in QE and APE using pretrained models
- Statistical models still show good performance
- Standard metrics may differ from industry established measurements
- Trade-offs are needed when considering what QE system to employ in practice
- QE aids APE.
- Not fine-tuning Transformer models is not better

- Future work:
  - Fine-tune BERT and XLM
  - Involve professional human translators
  - QE-based data decision

- General QE vs linguistic-phenomenon specific

Thank you for your attention