Robust and Scalable Cross-Lingual Transfer For Natural Language Understanding

Fabian David Schmidt

fdschmidt93.github.io

27.09.2023

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Fabian David Schmidt

- B.Sc. In Finance from Frankfurt School of Finance & Management
- Internships in Investment Banking & Private Equity (e.g., Goldman Sachs)

- M.Sc. In Data Science from University of Mannheim
- ML Engineering Internship at Car InsurTech Start-Up (Friday Versicherung)

- Third-year PhD in Cross-Lingual Representation Learning
- Open Source Enthusiast around Neovim ecosystem (telescope.nvim co-maintainer)

Making Sure We Are All On The Same Page!
Quick, Boring, But Important Preliminaries

- **ZS-XLT:** train XLM-R-{B,L}[1] on default English training sets by task, transfer without annotations to target languages
- **FS-XLT:** take model from ZS-XLT and adapt to target-language with a few (hundred) labelled target-language instances
- **Hyperparameters:** LR: 2e-5, batch size: 32, 10% linear warmup & decay
- **Tasks:**
  - **NLI:** determine whether "hypothesis" is true, false, or undetermined given a "premise"
  - **NER:** sequence-labelling task to predict whether & what named entity a token belongs to
  - **TyDiQA:** extractive QA, question answered by a span in given paragraph

1 XLM-R-base unless stated otherwise

Walkthrough

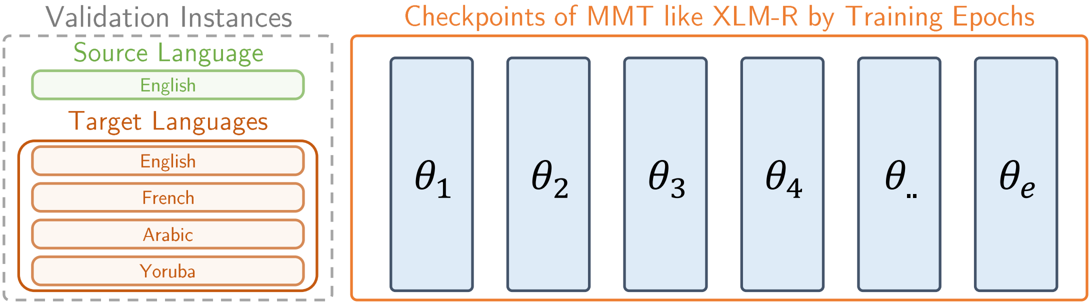
1. **Motivation:** Relevant Work overstates actionable XLT performance
2. **How Can We Strengthen XLT In Various Scenarios?**
    1. 'SLICER': Lever Task-Specific Properties for ZS-XLT in NER
    2. 'Don't Stop Fine-Tuning': Ground FS-XLT in Source-Language Data
    3. 'Free Lunch': More Robust {ZS,FS}-XLT With Simple Model Averaging
    4. One For All & All For One: Cumulative Averaging For Ideal ZS-XLT

XLT: cross-lingual transfer

ZS-XLT: zero-shot XLT; only fine-tune XLM-R/mT5 on English training data & transfer to target languages

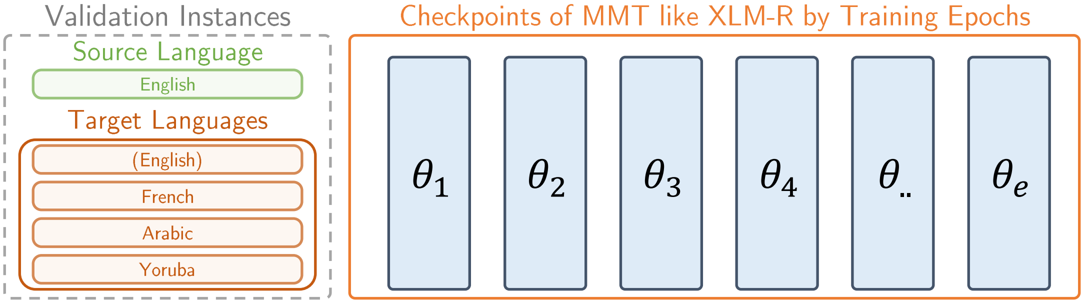
FS-XLT: few-shot XLT; like ZS-XLT, but further train on few (hundred) labelled target-language instances before transfer

Background: Model Selection in Cross-Lingual Transfer



- **Models:** finetune pretrained massively multilingual transformers like mT5 / XLM-R
- **Data:** train on sizable English task data & transfer zero- or few-shot to target languages

Background: Model Selection in Cross-Lingual Transfer



LAST

LAST simply picks the final checkpoint θ_e for testing

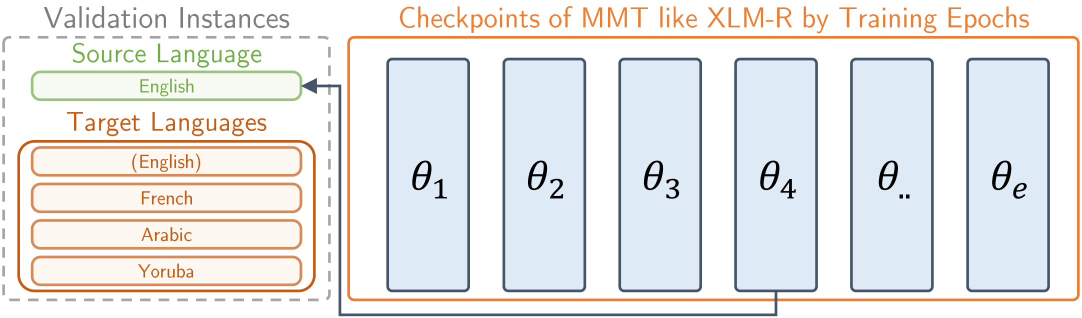
SRC-DEV

θ_4 maximizes source-language validation performance and is used for testing

TRG-DEV

θ_3 maximises avg. target-language validation performance (used by mT5, VECO)

Background: Model Selection in Cross-Lingual Transfer



LAST

LAST simply picks the final checkpoint θ_e for testing

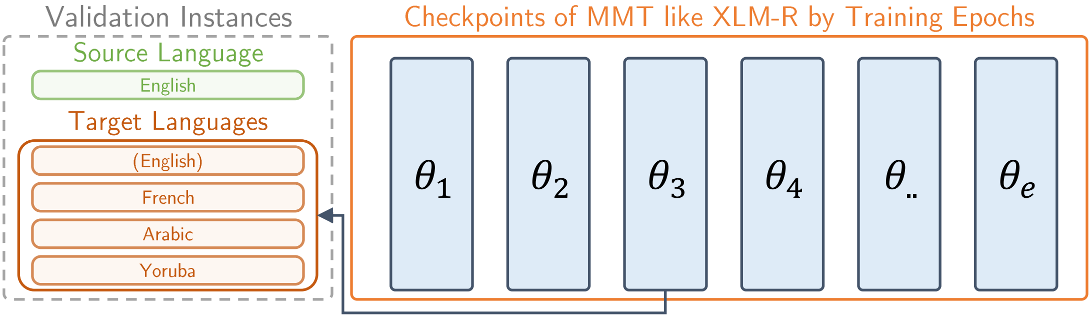
SRC-DEV

θ_4 maximizes source-language validation performance and is used for testing

TRG-DEV

θ_3 maximises avg. target-language validation performance (used by mT5, VECO)

Background: Model Selection in Cross-Lingual Transfer



LAST

LAST simply picks the final checkpoint θ_e for testing

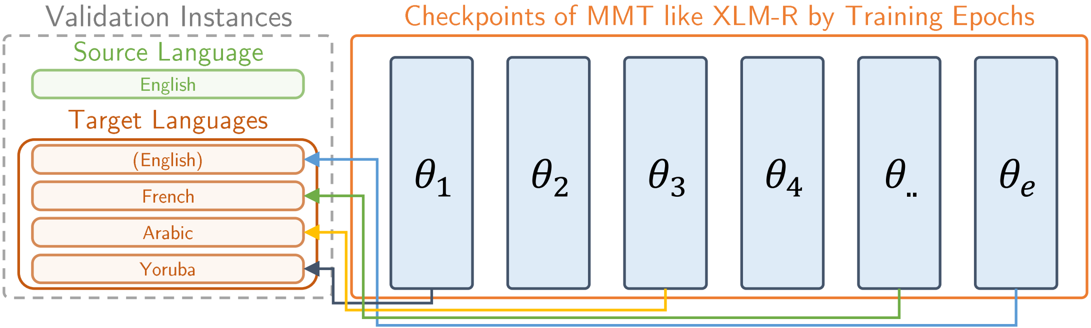
SRC-DEV

θ_4 maximizes source-language validation performance and is used for testing

TRG-DEV-AVG

θ_3 maximises avg. target-language validation performance (used by mT5, VECO)

Background: Model Selection in Cross-Lingual Transfer



LAST

LAST simply picks the final checkpoint θ_e for testing

SRC-DEV

θ_4 maximizes source-language validation performance and is used for testing

TRG-DEV-LS

Use target-language specific θ, denominates upper ZS-XLT bound (used for analysis)

Background: Model Selection in Cross-Lingual Transfer
TRG-DEV unrealistic in both Zero- and Few-Shot XLT

Fair	Problematic
LAST: LAST simply picks the final checkpoint θ_e for testing	**TRG-DEV-AVG**: θ_3 maximises avg. target-language validation performance (used by mT5, VECO)
SRC-DEV: θ_4 maximizes source-language validation performance and is used for testing	**TRG-DEV-LS**: Use target-language specific θ, denominates upper ZS-XLT bound (we use for analysis)

- **Opaque:** experimental setups in relevant work frequently underspecified (mT5, XLM-R)
- **Inefficient**: few hundred TRG-DEV instances better used for training!
- **Impractical:** TRG-DEV does not represent actionable XLT performance
- **Inconsistent:** TRG-DEV does not consistently reduce std. dev over LAST or S-DEV

Don't Use English Dev: On the Zero-Shot Cross-Lingual Evaluation of Contextual Embeddings
Don't Stop Fine-Tuning: On Training Regimes for Few-Shot Cross-Lingual Transfer with Multilingual Language Models
Free Lunch: Robust Cross-Lingual Transfer via Model Checkpoint Averaging

Background: Model Selection in Cross-Lingual Transfer

TRG-DEV unrealistic in both Zero- and Few-Shot XLT

Fair	Problematic
LAST LAST simply picks the final checkpoint θ_e for testing	**TRG-DEV-AVG** θ_3 maximises avg. target-language validation performance (used by mT5, VECO)
SRC-DEV θ_4 maximizes source-language validation performance and is used for testing	**TRG-DEV-LS** Use target-language specific θ, denominates upper ZS-XLT bound (we use for analysis)

- **Opaque:** experimental setups in relevant work frequently underspecified (mT5, XLM-R)
- **Inefficient**: few hundred TRG-DEV instances better used for training!
- **Impractical:** TRG-DEV does not represent actionable XLT performance
- **Inconsistent:** TRG-DEV does not consistently reduce std. dev over LAST or S-DEV

How do we optimize XLT without TRG-DEV?

Walkthrough

1. **Motivation:** Relevant Work overstates actionable XLT performance
2. **How Can We Strengthen XLT In Various Scenarios?**
    1. 'SLICER': Lever Task-Specific Properties for ZS-XLT in NER
    2. 'Don't Stop Fine-Tuning': Ground FS-XLT in Source-Language Data
    3. 'Free Lunch': More Robust {ZS,FS}-XLT With Simple Model Averaging
    4. One For All & All For One: Cumulative Averaging For Ideal ZS-XLT

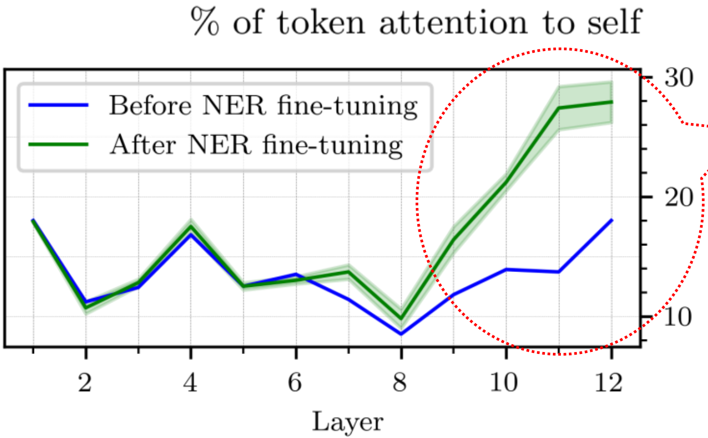
XLT: cross-lingual transfer

ZS-XLT: zero-shot XLT; only fine-tune XLM-R/mT5 on English training data & transfer to target languages

FS-XLT: few-shot XLT; like ZS-XLT, but further train on few (hundred) labelled target-language instances before transfer

Decontextualization in NER Impairs ZS-XLT

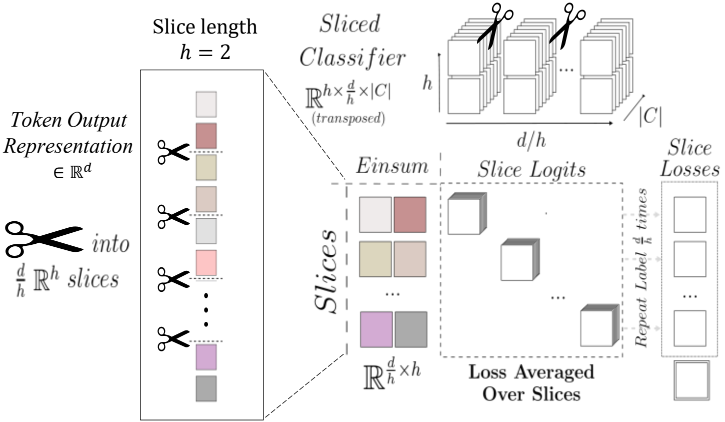
Analysis of Pre- and Post-Fine-Tuning of XLM-R base on WikiANN-EN train



- **% of token attention to self:** what attention probability does token pay to itself at layer N
- **Decontextualization in Fine-Tuning:** tokens attend more to themselves after fine-tuning!
- **(Monolingual) NER (over-)fits on token-specific information (casing, etc.):** "Obama" (New York City) is always a Person (City)!

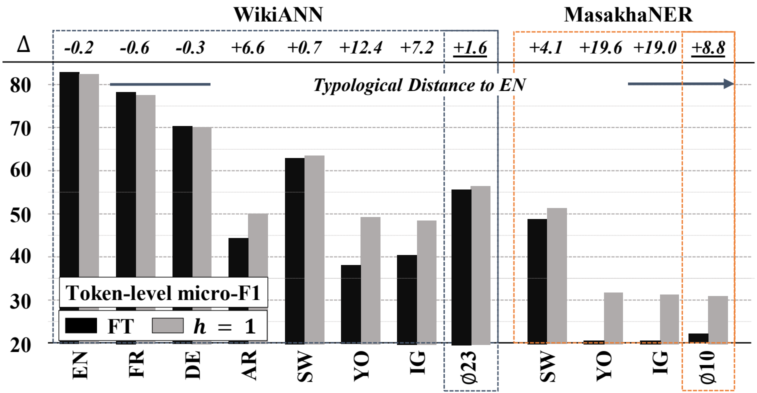
Solution: SLICER ✂ Token-Specific (Over-)fitting

Sliced Fine-Tuning For NER – Inference unchanged!



- ✂ along $\mathbb{R}^d$ of token and classifier representations
- **Loss:** Avg of token × #slices losses
- **Token-specific features don't fit into slice!**
- **Slices within tokens cannot share features!**
- **Inference:** additive ensemble over slices!

SLICER Improves ZS-XLT To Low-Resource Languages



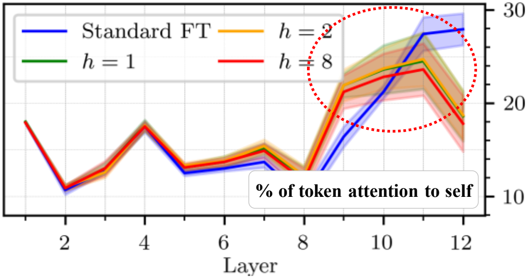
- **Setup:**
  - XLM-R base
  - Train on WikiANN-EN
  - Transfer after 10 epochs
- **Results robust for**
  (a) different hyperparameters
  (b) different source language (RU)
  (c) TRG-DEV: improves analogous to LAST at slightly slimmer margins

Decontextualization in NER SLICED for Better ZS-XLT

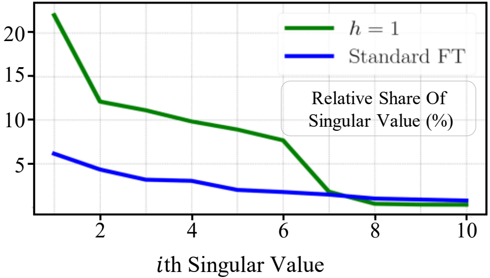
SLICER curbs Token Self-Attention & Favors Low-Rank Solutions

- SLICER forcefully decreases token dissimilarity by removing token-specific features
- Less token dissimilarity means higher NE-agnostic similarity to increase contextualization (i.e., forced attention to context), which coincides with lower-rank token embeddings

Token-Self Attention
for varying slice lengths h



SVD on Stacked Token Representations
of WikiANN-EN-test



SLICER Retrospective
One Of The "Quick" Ideas That Just Worked

- **Idea Origination:**
  - Previous work demonstrated L2-regularization slightly but consistently benefits ZS-XLT (with mBERT)
  - What about arbitrarily large dropout (before classifier)?
    - No Impact until 98%+ dropout on NER trials
      (did not test other tasks at the time)
  - SLICER ensures each dimension has to try to separate NER tokens
    (students found single dimension in $h = 1$ separates one-vs-all, as expected)

- **Does it work in practice?**
  - SLICER's benefits correlate well with how challenging transfer is!
    WikiANN to MasakhaNER is (a) cross-lingual & (b) cross-domain

Walkthrough

1. **Motivation:** Relevant Work overstates actionable XLT performance
2. **How Can We Strengthen XLT In Various Scenarios?**
   1. 'SLICER': Lever Task-Specific Properties for ZS-XLT in NER
   2. 'Don't Stop Fine-Tuning': Ground FS-XLT in Source-Language Data
   3. 'Free Lunch': More Robust {ZS,FS}-XLT With Simple Model Averaging
   4. One For All & All For One: Cumulative Averaging For Ideal ZS-XLT

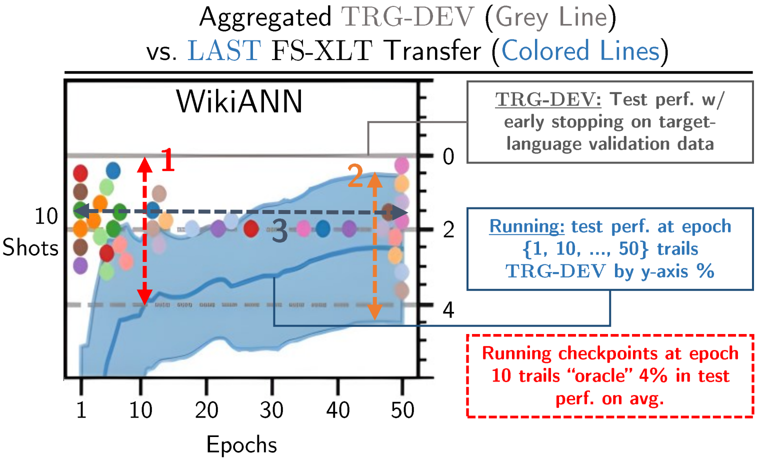
XLT: cross-lingual transfer

ZS-XLT: zero-shot XLT; only fine-tune XLM-R/mT5 on English training data & transfer to target languages

FS-XLT: few-shot XLT; like ZS-XLT, but further train on few (hundred) labelled target-language instances before transfer

Reliable Sequential Few-Shot Transfer Requires TRG-DEV

Three Major Problems With Sequential FS-XLT

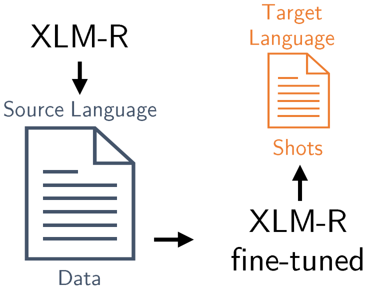


1. Large performance spread between TRG-DEV and true FS-XLT on average
2. Sizable fluctuations around true FS-XLT
3. Best TRG-DEV checkpoints are scattered (dots group target language by colour; each dot ran on 10 different shots)

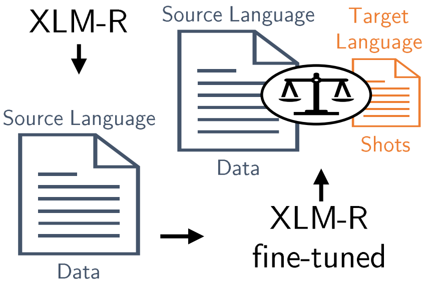
Simple Solution: Ground FS-XLT in Source Language Data

Reusing Source-Language Training Instances Improves FS-XLT

Before: Sequential Few-Shot Transfer

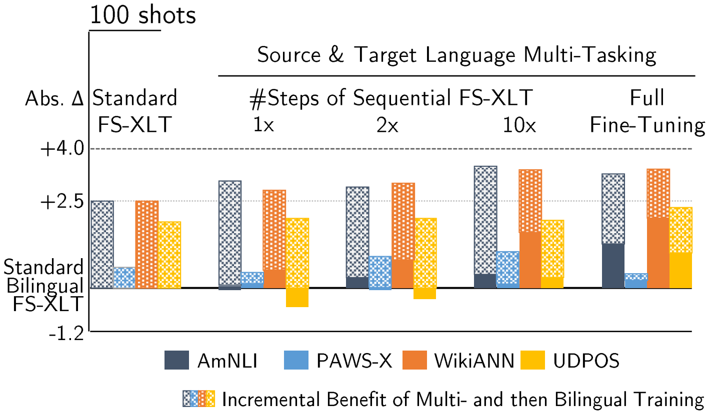


Now: Multi-Tasking on Source & Target Language



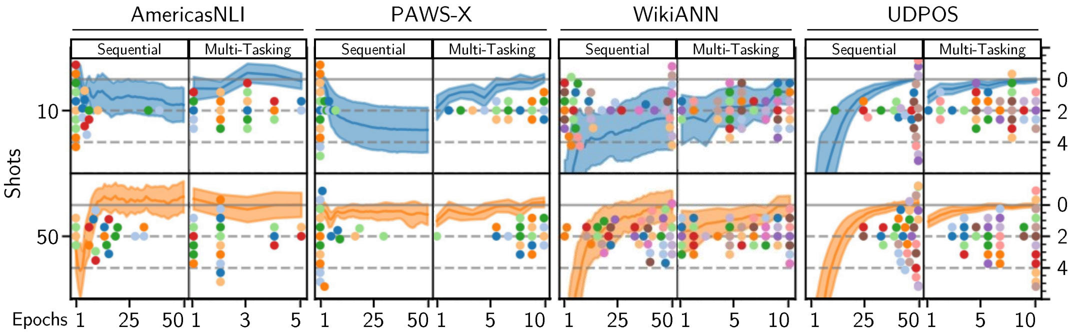
Source-Target Language Multi-Tasking Benefits FS-XLT

Consistent Performance Gains.. (1/2)



- **Setup:** True Few-Shot Transfer
  - w/o target language validation data
  - fixed hyperparameters
- **Multi-Tasking** outperforms with ↑steps
- **Intermediate Multilingual Fine-Tuning** (EN → MULTI → TRG-LANG) improves computational efficiency & performance

Source-Target Language Multi-Tasking Benefits FS-XLT
..that we can seize upon reliably without TRG-DEV (2/2)



- **Performance:** Multi-Tasking on par or better than TRG-DEV in True Transfer ("LAST")
- **Consistency:** Best Transfer consistently at final epoch

‘Don’t Stop Fine-Tuning’ Retrospective
Simplicity over Complexity (1/2)

- **Idea Origination:**
  - Mix-Up works very well in computer vision and makes sense for FS-XLT
    - No improvements from mix-up; S&T multi-tasking was ideally only ablation but killed mix-up instead

- **Does it work in practice?** Yes, with some caveats
  - Multi-tasking detrimental in cross-lingual, cross-domain transfer, e.g. WikiANN & MasakhaNER
  - Effects generally diminish in higher resource setups
    - AND: Learning rate schedule with linear warm up and decay gets you very close to S&T multi-tasking on *some* tasks: multi-tasking provides safety
  - S&T multi-tasking helps regularizing (ZH translations as a source language observed more benefits)

Walkthrough

1. **Motivation:** Relevant Work overstates actionable XLT performance
2. **How Can We Strengthen XLT In Various Scenarios?**
   1. 'SLICER': Lever Task-Specific Properties for ZS-XLT in NER
   2. 'Don't Stop Fine-Tuning': Ground FS-XLT in Source-Language Data
   3. 'Free Lunch': More Robust {ZS,FS}-XLT With Simple Model Averaging
   4. One For All & All For One: Cumulative Averaging For Ideal ZS-XLT

XLT: cross-lingual transfer

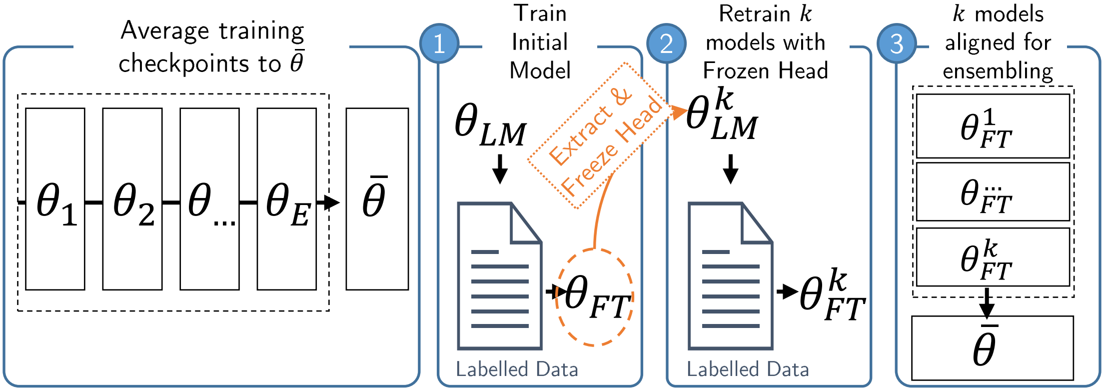
ZS-XLT: zero-shot XLT; only fine-tune XLM-R/mT5 on English training data & transfer to target languages

FS-XLT: few-shot XLT; like ZS-XLT, but further train on few (hundred) labelled target-language instances before transfer

Model Averaging For Robust XLT

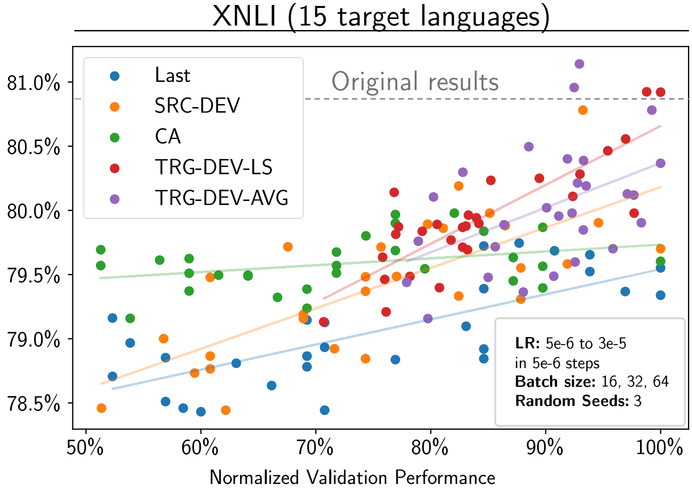
Single Run: Checkpoint Averaging (CA)

Multiple Runs: Aligned Heads Enable Ensembling via 'Run Averaging (RA)'



Model Selection in Zero-Shot Cross-Lingual Transfer (1/2)

ZS-XLT on EN-trained XLM-R-L on 3x 21 pairs of (LR, batch size)

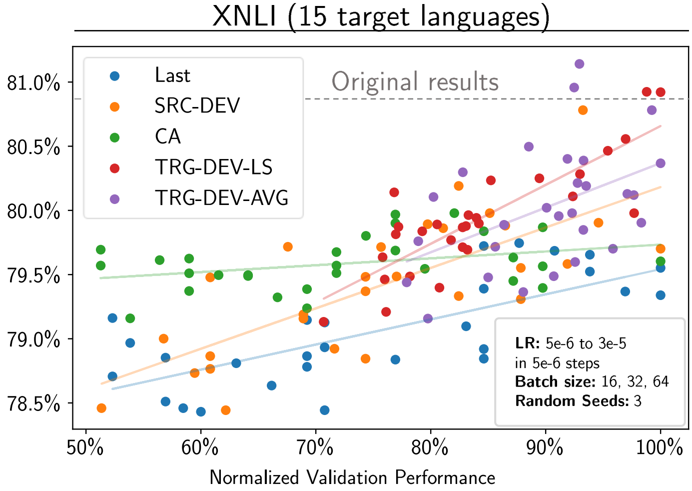


- **X-axis:** Top 50% relative to best val. LAST, ..., TRG-DEV-AVG shown
- **Y-axis:** Avg. test perf. on all target languages

- **LAST, SRC-DEV,** and **CA** are measured by **source-language validation** performance
- **TRG-DEV-LS:** select checkpoints for each target-language individually on its dev. set
- **TRG-DEV-AVG:** select single checkpoint for all target languages on avg. target-language validation performance

Model Selection in Zero-Shot Cross-Lingual Transfer (1/2)

ZS-XLT on EN-trained XLM-R-L on 3x 21 pairs of (LR, batch size)

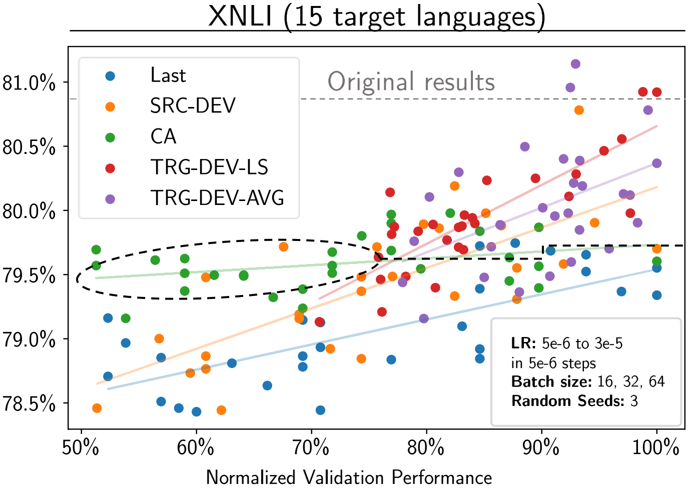


- **X-axis:** Top 50% relative to best val. LAST, ..., TRG-DEV-AVG shown
- **Y-axis:** Avg. test perf. on all target languages

- Hyperparameters matter (upward linear fits)

Model Selection in Zero-Shot Cross-Lingual Transfer (1/2)

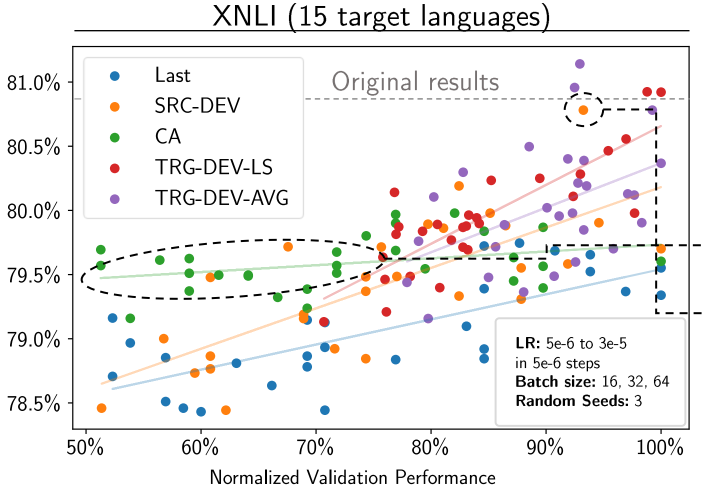
ZS-XLT on EN-trained XLM-R-L on 3x 21 pairs of (LR, batch size)



- **X-axis:** Top 50% relative to best val. LAST, ..., TRG-DEV-AVG shown
- **Y-axis:** Avg. test perf. on all target languages

- Hyperparameters matter (upward linear fits)
- **CA** much stronger at less- than-ideal hyperparameters

Model Selection in Zero-Shot Cross-Lingual Transfer (1/2)
ZS-XLT on EN-trained XLM-R-L on 3x 21 pairs of (LR, batch size)

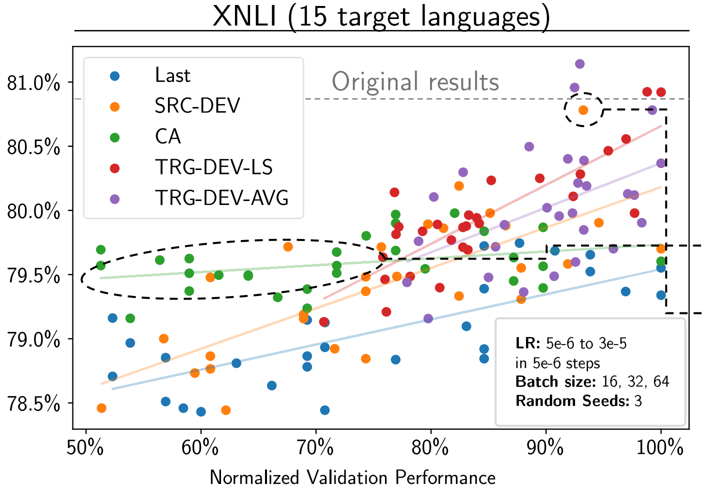


- **X-axis:** Top 50% relative to best val. LAST, ..., TRG-DEV-AVG shown
- **Y-axis:** Avg. test perf. on all target languages

- Hyperparameters matter (upward linear fits)
- **CA** much stronger at less- than-ideal hyperparameters
- **SRC-DEV:** 1 in 63 runs reproduces original perf. (only 3rd in val. perf.!)

Model Selection in Zero-Shot Cross-Lingual Transfer (1/2)

ZS-XLT on EN-trained XLM-R-L on 3x 21 pairs of (LR, batch size)

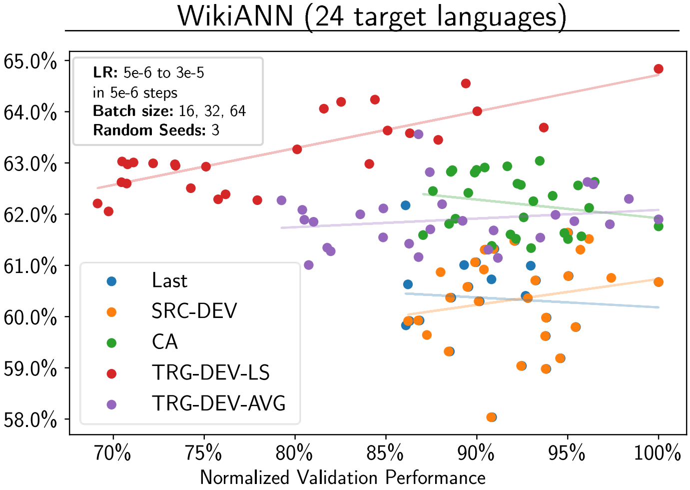


- **X-axis:** Top 50% relative to best val. LAST, ..., TRG-DEV-AVG shown
- **Y-axis:** Avg. test perf. on all target languages

- Hyperparameters matter (upward linear fits)
- **CA** much stronger at less- than-ideal hyperparameters
- **SRC-DEV:** 1 in 63 runs reproduces original perf. (only 3rd in val. perf.!)
- **TRG-DEV-AVG** strong in higher-level semantic tasks; only small gains via **TRG-DEV-LS** (again recall that training on TRG-DEV always better)

Model Selection in Zero-Shot Cross-Lingual Transfer (2/2)

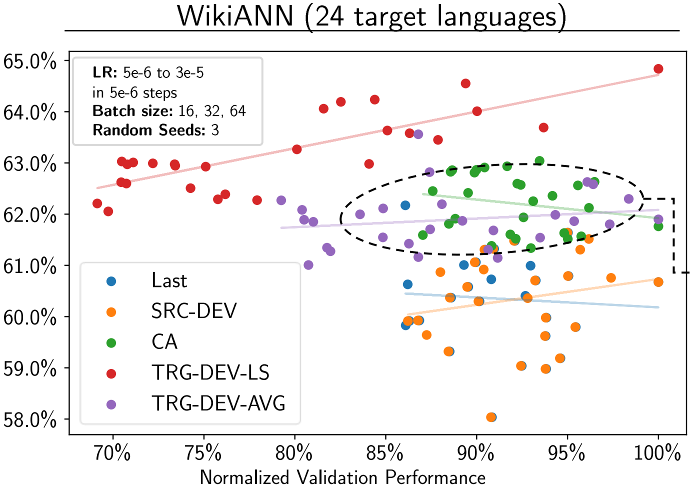
ZS-XLT on EN-trained XLM-R-L on 3x 21 pairs of (LR, batch size)



- **X-axis:** Top 50% relative to best val. LAST, ..., TRG-DEV-AVG shown
- **Y-axis:** Avg. test perf. on all target languages

- Much less/worse correlation with source-language validation perf. (downward linear fit)

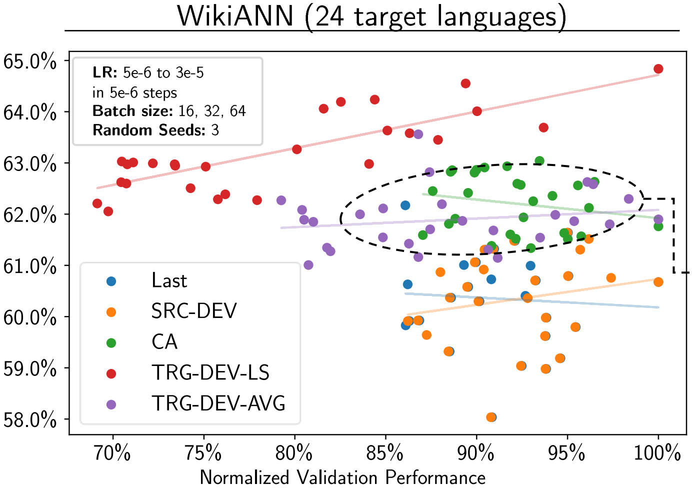
Model Selection in Zero-Shot Cross-Lingual Transfer (2/2)
ZS-XLT on EN-trained XLM-R-L on 3x 21 pairs of (LR, batch size)



- **X-axis:** Top 50% relative to best val. LAST, ..., TRG-DEV-AVG shown
- **Y-axis:** Avg. test perf. on all target languages

- Much less/worse correlation with source-language validation perf. (downward linear fit)
- **CA** even frequently beats **TRG-DEV-AVG**

Model Selection in Zero-Shot Cross-Lingual Transfer (2/2)
ZS-XLT on EN-trained XLM-R-L on 3x 21 pairs of (LR, batch size)



- **X-axis:** Top 50% relative to best val. LAST, ..., TRG-DEV-AVG shown
- **Y-axis:** Avg. test perf. on all target languages

- Much less/worse correlation with source-language validation perf. (downward linear fit)
- **CA** even frequently beats **TRG-DEV-AVG**
- **TRG-DEV-LS** successfully captures language-specific variation for strongest transfer performance

Can We ‘Fairly’ (without TRG-DEV) achieve ideal ZS-XLT?

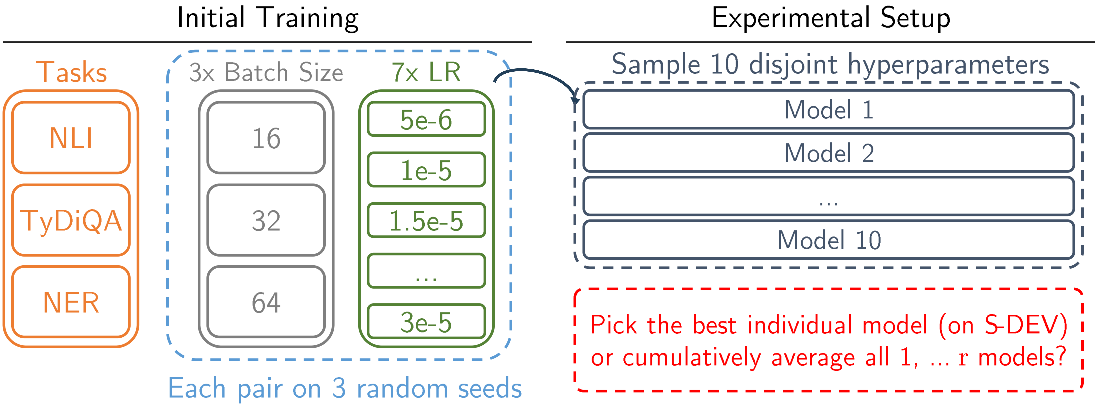
Can We ‘Fairly’ (without TRG-DEV) achieve ideal ZS-XLT?

Yes! In Surprisingly Naïve Manner

Model Selection vs. Cumulative Averaging for ZS-XLT

XLM-R-Large on a broad grid of hyperparameters

- **Again:** 61 runs over large task-agnostic grid of 21 pairs of learning rates and batch sizes, each for 3 seeds
- **Now:** Cumulatively sample 1,..., 10 runs with **disjoint tuples of learning rate and batch size** (10×) and



Model Selection vs. Cumulative Averaging for ZS-XLT

XLM-R-Large on a broad grid of hyperparameters

- **Again:** 61 runs over large task-agnostic grid of 21 pairs of learning rates and batch sizes, each for 3 seeds
- **Now:** Cumulatively sample 1,..., 10 runs with **disjoint tuples of learning rate and batch size** (10×) and
  - **Max. SRC-DEV**: pick best individual model on SRC-DEV for model variant LAST, SRC-DEV, CA
  - Cumulatively average all available LAST, SRC-DEV, CA model variants **without** SRC-DEV

	NLI						TyDiQA-GoldP						NER					
	Max. SRC-DEV			Cumulative Averaging			Max. SRC-DEV			Cumulative Averaging			Max. SRC-DEV			Cumulative Averaging		
r	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA
1	$76.5_{0.6}$	$76.5_{0.8}$	$77.3_{0.4}$	$76.5_{0.6}$	$76.5_{0.8}$	$77.3_{0.4}$	$71.9_{0.4}$	$71.9_{0.7}$	$73.6_{1.9}$	$71.9_{0.4}$	$71.9_{0.7}$	$73.6_{1.9}$	**$40.8_{2.7}$**	$41.1_{3.1}$	**$44.6_{2.1}$**	$40.8_{2.7}$	$41.1_{3.0}$	$44.6_{2.1}$
2	$77.2_{0.3}$	$77.5_{0.4}$	**$77.6_{0.2}$**	$77.6_{0.3}$	$77.8_{0.4}$	$78.0_{0.2}$	$71.9_{0.6}$	$71.6_{0.6}$	$73.3_{2.0}$	$73.4_{1.2}$	$73.3_{1.1}$	$72.9_{2.8}$	$39.3_{2.1}$	$39.3_{2.1}$	$43.5_{1.1}$	$43.2_{2.2}$	$43.2_{2.2}$	$45.6_{1.5}$
3	$77.2_{0.3}$	$77.5_{0.4}$	$77.6_{0.2}$	$77.8_{0.3}$	$77.9_{0.4}$	$78.1_{0.2}$	$72.1_{0.8}$	$71.8_{0.8}$	$74.1_{1.0}$	$74.1_{0.7}$	$74.2_{0.7}$	$73.8_{1.2}$	$39.3_{1.2}$	$39.5_{1.7}$	$44.0_{1.1}$	$45.0_{1.7}$	$45.1_{1.8}$	$47.3_{1.3}$
4	$77.2_{0.4}$	$77.5_{0.4}$	$77.5_{0.2}$	$77.7_{0.3}$	$77.9_{0.4}$	$78.1_{0.3}$	$72.5_{0.8}$	**$72.0_{0.9}$**	$73.9_{0.6}$	$74.5_{0.6}$	$74.1_{0.4}$	$74.1_{0.9}$	$40.2_{2.0}$	$40.8_{2.2}$	$44.5_{1.5}$	$45.0_{1.7}$	$45.3_{1.8}$	$47.2_{1.4}$
5	$77.3_{0.3}$	**$77.6_{0.3}$**	$77.5_{0.2}$	$77.9_{0.2}$	$78.0_{0.2}$	$78.1_{0.1}$	**$72.6_{0.8}$**	$72.0_{0.9}$	$73.8_{0.8}$	**$74.7_{0.7}$**	**$74.4_{0.6}$**	$74.2_{0.8}$	$40.3_{2.0}$	**$41.2_{2.3}$**	$43.9_{1.9}$	$45.3_{1.7}$	$45.5_{1.7}$	$47.5_{1.4}$
6	$77.3_{0.3}$	$77.6_{0.1}$	$77.5_{0.2}$	$77.9_{0.1}$	$78.0_{0.2}$	$78.1_{0.1}$	$72.6_{0.8}$	$72.0_{0.9}$	$74.2_{0.5}$	$74.7_{0.7}$	$74.4_{0.5}$	$74.2_{0.7}$	$40.3_{2.0}$	$41.2_{2.3}$	$43.9_{1.9}$	$45.7_{1.4}$	$45.9_{1.4}$	$47.9_{1.2}$
7	$77.3_{0.3}$	$77.6_{0.1}$	$77.5_{0.2}$	$77.9_{0.2}$	$78.1_{0.2}$	$78.2_{0.2}$	$72.3_{0.9}$	$71.7_{0.7}$	**$74.3_{0.3}$**	$74.6_{0.7}$	$74.3_{0.6}$	$74.2_{0.5}$	$40.0_{2.1}$	$40.6_{2.3}$	$44.1_{2.0}$	$46.0_{1.3}$	$46.1_{1.3}$	$48.1_{1.1}$
8	$77.3_{0.3}$	$77.6_{0.2}$	$77.5_{0.2}$	**$78.0_{0.2}$**	**$78.2_{0.1}$**	**$78.3_{0.2}$**	$72.1_{0.9}$	$71.7_{0.7}$	$74.2_{0.5}$	$74.6_{0.7}$	$74.3_{0.5}$	**$74.3_{0.5}$**	$40.0_{2.1}$	$40.6_{2.3}$	$44.6_{1.7}$	$46.0_{1.1}$	$46.1_{1.2}$	$48.2_{1.0}$
9	**$77.4_{0.2}$**	$77.6_{0.2}$	$77.6_{0.2}$	$78.0_{0.1}$	$78.1_{0.1}$	$78.3_{0.2}$	$72.0_{1.1}$	$71.7_{0.7}$	$74.2_{0.5}$	$74.6_{0.5}$	$74.4_{0.4}$	$74.2_{0.4}$	$39.6_{2.3}$	$39.9_{2.4}$	$44.3_{1.8}$	$46.0_{0.6}$	$46.1_{0.7}$	$48.3_{0.7}$
10	$77.3_{0.2}$	$77.6_{0.2}$	$77.6_{0.2}$	$78.0_{0.1}$	$78.2_{0.1}$	$78.3_{0.1}$	$72.0_{1.1}$	$71.7_{0.7}$	$74.2_{0.5}$	$74.6_{0.6}$	$74.4_{0.4}$	$74.2_{0.6}$	$39.6_{2.3}$	$39.9_{2.4}$	$44.4_{1.7}$	**$46.1_{0.5}$**	**$46.2_{0.6}$**	**$48.4_{0.5}$**

Model Selection vs. Cumulative Averaging for ZS-XLT

XLM-R-Large on a broad grid of hyperparameters

- **Again:** 61 runs over large task-agnostic grid of 21 pairs of learning rates and batch sizes, each for 3 seeds
- **Now:** Cumulatively sample 1,..., 10 runs with **disjoint tuples of learning rate and batch size** (10×) and
  - **Max. SRC-DEV**: pick best individual model on SRC-DEV for model variant LAST, SRC-DEV, CA
  - Cumulatively average all available LAST, SRC-DEV, CA model variants **without** SRC-DEV

	NLI						TyDiQA-GoldP						NER					
	Max. SRC-DEV			Cumulative Averaging			Max. SRC-DEV			Cumulative Averaging			Max. SRC-DEV			Cumulative Averaging		
r	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA
1	$76.5_{0.6}$	$76.5_{0.8}$	$77.3_{0.4}$	$76.5_{0.6}$	$76.5_{0.8}$	$77.3_{0.4}$	$71.9_{0.4}$	$71.9_{0.7}$	$73.6_{1.9}$	$71.9_{0.4}$	$71.9_{0.7}$	$73.6_{1.9}$	**$40.8_{2.7}$**	$41.1_{3.1}$	**$44.6_{2.1}$**	$40.8_{2.7}$	$41.1_{3.0}$	$44.6_{2.1}$
2	$77.2_{0.3}$	$77.5_{0.4}$	**$77.6_{0.2}$**	$77.6_{0.3}$	$77.8_{0.4}$	$78.0_{0.2}$	$71.9_{0.6}$	$71.6_{0.6}$	$73.3_{2.0}$	$73.4_{1.2}$	$73.3_{1.1}$	$72.9_{2.8}$	$39.3_{2.1}$	$39.3_{2.1}$	$43.5_{1.1}$	$43.2_{2.2}$	$43.2_{2.2}$	$45.6_{1.5}$
3	$77.2_{0.3}$	$77.5_{0.4}$	$77.6_{0.2}$	$77.8_{0.3}$	$77.9_{0.4}$	$78.1_{0.2}$	$72.1_{0.8}$	$71.8_{0.8}$	$74.1_{1.0}$	$74.1_{0.7}$	$74.2_{0.7}$	$73.8_{1.2}$	$39.3_{1.2}$	$39.5_{1.7}$	$44.0_{1.1}$	$45.0_{1.7}$	$45.1_{1.8}$	$47.3_{1.3}$
4	$77.2_{0.4}$	$77.5_{0.4}$	$77.5_{0.2}$	$77.7_{0.3}$	$77.9_{0.4}$	$78.1_{0.3}$	$72.5_{0.8}$	**$72.0_{0.9}$**	$73.9_{0.6}$	$74.5_{0.6}$	$74.1_{0.4}$	$74.1_{0.9}$	$40.2_{2.0}$	$40.8_{2.2}$	$44.5_{1.5}$	$45.0_{1.7}$	$45.3_{1.8}$	$47.2_{1.4}$
5	$77.3_{0.3}$	**$77.6_{0.3}$**	$77.5_{0.2}$	$77.9_{0.2}$	$78.0_{0.2}$	$78.1_{0.1}$	**$72.6_{0.8}$**	$72.0_{0.9}$	$73.8_{0.8}$	**$74.7_{0.7}$**	**$74.4_{0.6}$**	$74.2_{0.8}$	$40.3_{2.0}$	**$41.2_{2.3}$**	$43.9_{1.9}$	$45.3_{1.7}$	$45.5_{1.7}$	$47.5_{1.4}$
6	$77.3_{0.3}$	$77.6_{0.1}$	$77.5_{0.2}$	$77.9_{0.1}$	$78.0_{0.2}$	$78.1_{0.1}$	$72.6_{0.8}$	$72.0_{0.9}$	$74.2_{0.5}$	$74.7_{0.7}$	$74.4_{0.5}$	$74.2_{0.7}$	$40.3_{2.0}$	$41.2_{2.3}$	$43.9_{1.9}$	$45.7_{1.4}$	$45.9_{1.4}$	$47.9_{1.2}$
7	$77.3_{0.3}$	$77.6_{0.1}$	$77.5_{0.2}$	$77.9_{0.2}$	$78.1_{0.2}$	$78.2_{0.2}$	$72.3_{0.9}$	$71.7_{0.7}$	**$74.3_{0.3}$**	$74.6_{0.7}$	$74.3_{0.6}$	$74.2_{0.5}$	$40.0_{2.1}$	$40.6_{2.3}$	$44.1_{2.0}$	$46.0_{1.3}$	$46.1_{1.3}$	$48.1_{1.1}$
8	$77.3_{0.3}$	$77.6_{0.2}$	$77.5_{0.2}$	**$78.0_{0.2}$**	**$78.2_{0.1}$**	**$78.3_{0.2}$**	$72.1_{0.9}$	$71.7_{0.7}$	$74.2_{0.5}$	$74.6_{0.7}$	$74.3_{0.5}$	**$74.3_{0.5}$**	$40.0_{2.1}$	$40.6_{2.3}$	$44.6_{1.7}$	$46.0_{1.1}$	$46.1_{1.2}$	$48.2_{1.0}$
9	**$77.4_{0.2}$**	$77.6_{0.2}$	$77.6_{0.2}$	$78.0_{0.1}$	$78.1_{0.1}$	$78.3_{0.2}$	$72.0_{1.1}$	$71.7_{0.7}$	$74.2_{0.5}$	$74.6_{0.5}$	$74.4_{0.4}$	$74.2_{0.4}$	$39.6_{2.3}$	$39.9_{2.4}$	$44.3_{1.8}$	$46.0_{0.6}$	$46.1_{0.7}$	$48.3_{0.7}$
10	$77.3_{0.2}$	$77.6_{0.2}$	$77.6_{0.2}$	$78.0_{0.1}$	$78.2_{0.1}$	$78.3_{0.1}$	$72.0_{1.1}$	$71.7_{0.7}$	$74.2_{0.5}$	$74.6_{0.6}$	$74.4_{0.4}$	$74.2_{0.6}$	$39.6_{2.3}$	$39.9_{2.4}$	$44.4_{1.7}$	**$46.1_{0.5}$**	**$46.2_{0.6}$**	**$48.4_{0.5}$**

Model Selection vs. Cumulative Averaging for ZS-XLT

XLM-R-Large on a broad grid of hyperparameters

- **Again:** 61 runs over large task-agnostic grid of 21 pairs of learning rates and batch sizes, each for 3 seeds
- **Now:** Cumulatively sample 1,..., 10 runs with **disjoint tuples of learning rate and batch size** (10×) and
- Green denotes on par (light) or better (strong) performance than **best** max. SRC-DEV (L, S-DEV, CA)
- Cumulative averaging typically achieves better performance from first averaged-in run ($r = 2$) at lower σ

	NLI						TyDiQA-GoldP						NER					
	Max. SRC-DEV			Cumulative Averaging			Max. SRC-DEV			Cumulative Averaging			Max. SRC-DEV			Cumulative Averaging		
r	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA	LAST	S-DEV	CA
1	76.5$_{0.6}$	76.5$_{0.8}$	77.3$_{0.4}$	76.5$_{0.6}$	76.5$_{0.8}$	77.3$_{0.4}$	71.9$_{0.4}$	71.9$_{0.7}$	73.6$_{1.9}$	71.9$_{0.4}$	71.9$_{0.7}$	73.6$_{1.9}$	**40.8$_{2.7}$**	41.1$_{3.1}$	**44.6$_{2.1}$**	40.8$_{2.7}$	41.1$_{3.0}$	44.6$_{2.1}$
2	77.2$_{0.3}$	77.5$_{0.4}$	**77.6$_{0.2}$**	77.6$_{0.3}$	77.8$_{0.4}$	78.0$_{0.2}$	71.9$_{0.6}$	71.6$_{0.6}$	73.3$_{2.0}$	73.4$_{1.2}$	73.3$_{1.1}$	72.9$_{2.8}$	39.3$_{2.1}$	39.3$_{2.1}$	43.5$_{1.1}$	43.2$_{2.2}$	43.2$_{2.2}$	45.6$_{1.5}$
3	77.2$_{0.3}$	77.5$_{0.4}$	77.6$_{0.2}$	77.8$_{0.3}$	77.9$_{0.4}$	78.1$_{0.2}$	72.1$_{0.8}$	71.8$_{0.8}$	74.1$_{1.0}$	74.1$_{0.7}$	74.2$_{0.7}$	73.8$_{1.2}$	39.3$_{1.2}$	39.5$_{1.7}$	44.0$_{1.1}$	45.0$_{1.7}$	45.1$_{1.8}$	47.3$_{1.3}$
4	77.2$_{0.4}$	77.5$_{0.4}$	77.5$_{0.2}$	77.7$_{0.3}$	77.9$_{0.4}$	78.1$_{0.3}$	72.5$_{0.8}$	**72.0$_{0.9}$**	73.9$_{0.6}$	74.5$_{0.6}$	74.1$_{0.4}$	74.1$_{0.9}$	40.2$_{2.0}$	40.8$_{2.2}$	44.5$_{1.5}$	45.0$_{1.7}$	45.3$_{1.8}$	47.2$_{1.4}$
5	77.3$_{0.3}$	**77.6$_{0.3}$**	77.5$_{0.2}$	77.9$_{0.2}$	78.0$_{0.2}$	78.1$_{0.1}$	**72.6$_{0.8}$**	72.0$_{0.9}$	73.8$_{0.8}$	**74.7$_{0.7}$**	**74.4$_{0.6}$**	74.2$_{0.8}$	40.3$_{2.0}$	**41.2$_{2.3}$**	43.9$_{1.9}$	45.3$_{1.7}$	45.5$_{1.7}$	47.5$_{1.4}$
6	77.3$_{0.3}$	77.6$_{0.1}$	77.5$_{0.2}$	77.9$_{0.1}$	78.0$_{0.2}$	78.1$_{0.1}$	72.6$_{0.8}$	72.0$_{0.9}$	74.2$_{0.5}$	74.7$_{0.7}$	74.4$_{0.5}$	74.2$_{0.7}$	40.3$_{2.0}$	41.2$_{2.3}$	43.9$_{1.9}$	45.7$_{1.4}$	45.9$_{1.4}$	47.9$_{1.2}$
7	77.3$_{0.3}$	77.6$_{0.1}$	77.5$_{0.2}$	77.9$_{0.2}$	78.1$_{0.2}$	78.2$_{0.2}$	72.3$_{0.9}$	71.7$_{0.7}$	**74.3$_{0.3}$**	74.6$_{0.7}$	74.3$_{0.6}$	74.2$_{0.5}$	40.0$_{2.1}$	40.6$_{2.3}$	44.1$_{2.0}$	46.0$_{1.3}$	46.1$_{1.3}$	48.1$_{1.1}$
8	77.3$_{0.3}$	77.6$_{0.2}$	77.5$_{0.2}$	**78.0$_{0.2}$**	**78.2$_{0.1}$**	**78.3$_{0.2}$**	72.1$_{0.9}$	71.7$_{0.7}$	74.2$_{0.5}$	74.6$_{0.7}$	74.3$_{0.5}$	**74.3$_{0.5}$**	40.0$_{2.1}$	40.6$_{2.3}$	44.6$_{1.7}$	46.0$_{1.1}$	46.1$_{1.2}$	48.2$_{1.0}$
9	**77.4$_{0.2}$**	77.6$_{0.2}$	77.6$_{0.2}$	78.0$_{0.1}$	78.1$_{0.1}$	78.3$_{0.2}$	72.0$_{1.1}$	71.7$_{0.7}$	74.2$_{0.5}$	74.6$_{0.5}$	74.4$_{0.4}$	74.2$_{0.4}$	39.6$_{2.3}$	39.9$_{2.4}$	44.3$_{1.8}$	46.0$_{0.6}$	46.1$_{0.7}$	48.3$_{0.7}$
10	77.3$_{0.2}$	77.6$_{0.2}$	77.6$_{0.2}$	78.0$_{0.1}$	78.2$_{0.1}$	78.3$_{0.1}$	72.0$_{1.1}$	71.7$_{0.7}$	74.2$_{0.5}$	74.6$_{0.6}$	74.4$_{0.4}$	74.2$_{0.6}$	39.6$_{2.3}$	39.9$_{2.4}$	44.4$_{1.7}$	**46.1$_{0.5}$**	**46.2$_{0.6}$**	**48.4$_{0.5}$**

Cumulative Avg. aligns with 'ideal' (TRG-DEV) ZS-XLT

	NLI				TyDiQA-GoldP				NER			
	Max. DEV		Cum. Avg.		Max. Dev		Cum. Avg.		Max. DEV		Cum. Avg.	
r	SRC DEV	TRG DEV	CA	SOUP	SRC DEV	TRG DEV	CA	SOUP	SRC DEV	TRG DEV	CA	SOUP
1	77.3	77.0	77.3	76.8	71.9	72.8	73.6	73.7	41.1	46.5	44.6	42.3
3	77.5	77.7	78.1	77.6	71.8	73.5	73.8	73.8	39.5	49.2	47.3	42.1
5	**77.6**	77.9	78.1	77.6	**72.0**	73.4	74.2	**74.3**	**41.2**	49.7	47.5	**42.8**
7	77.6	78.2	78.2	**77.8**	71.7	73.7	74.2	73.9	40.6	49.9	48.1	42.8
10	77.6	78.4	78.3	77.7	71.7	73.9	74.2	73.8	39.9	49.9	48.4	42.8

- Repeat prior analysis now with TRG-DEV and SOUP
- **SOUP** averages top-k SRC-DEV checkpoints), but plateaus like max. **SRC-DEV** (albeit at higher levels)

Cumulative Avg. aligns with 'ideal' (TRG-DEV) ZS-XLT

	NLI				TyDiQA-GoldP				NER			
	Max. DEV		Cum. Avg.		Max. Dev		Cum. Avg.		Max. DEV		Cum. Avg.	
r	SRC DEV	TRG DEV	CA	SOUP	SRC DEV	TRG DEV	CA	SOUP	SRC DEV	TRG DEV	CA	SOUP
1	77.3	77.0	77.3	76.8	71.9	72.8	73.6	73.7	41.1	46.5	44.6	42.3
3	77.5	77.7	78.1	77.6	71.8	73.5	73.8	73.8	39.5	49.2	47.3	42.1
5	77.6	77.9	78.1	77.6	72.0	73.4	**74.2**	74.3	41.2	49.7	47.5	42.8
7	77.6	78.2	78.2	77.8	71.7	73.7	74.2	73.9	40.6	**49.9**	48.1	42.8
10	77.6	**78.4**	**78.3**	77.7	71.7	**73.9**	74.2	73.8	39.9	49.9	**48.4**	42.8

- Repeat prior analysis now with TRG-DEV and SOUP
- SOUP averages top-k SRC-DEV checkpoints), but plateaus like max. SRC-DEV (albeit at higher levels)
- **Key:** naively cumulative averaging **without** monitoring SRC-DEV

Cumulative Averaging

1. irons out bad runs
2. ingests strong runs (cf. TRG-DEV)
3. does not plateau in sub-optimal SRC-DEV

‘Free Lunch’ and ‘One For All’ Retrospective
Simplicity over Complexity (2/2)

- **Idea Origination:**
  - Complex ideas around multi-lingual FS-XLT itched me the wrong way (gradient vaccination)
  - ‘Model soups’ (complex variant of run averaging) worked well in vision, and, turns out, checkpoint avg. very common for machine translation
  - Run averaging did not work in ‘model soups’: heads were aligned for CV (same induction) but not for text classification

- **Does it work in practice?**
  - Checkpoint averaging effects diminish for higher-resource FS-XLT setups (transfer becomes more ‘monolingual’)
    - translate-train would arguably be interesting to see
  - (Cumulative) Run Averaging Defensive Strategy That Gives You – with very high likelihood – best XLT performance (at same inference speed)

Universal Take-Aways

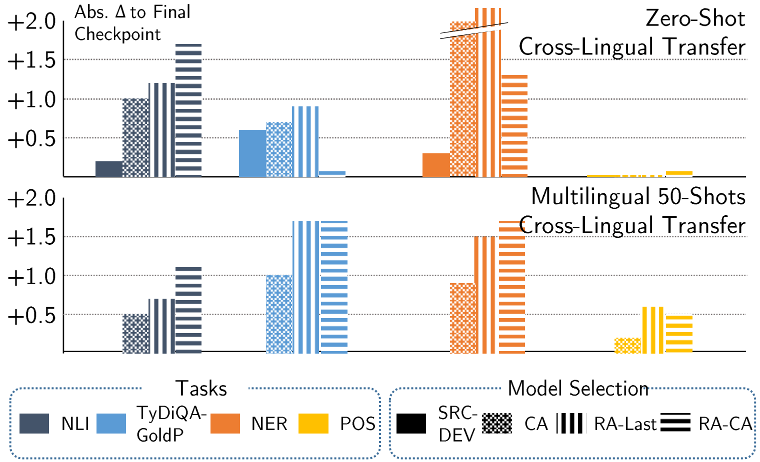
Irrespective of Cross-Lingual Transfer Evaluation

- Fairness in evaluation critically important for fundamental progress
- Simple methods can work just as well as more involved approaches
  - Model ‘averaging’ or ‘ensembles’ in various forms (RA, MoE) are very strong baselines if you are already tuning hyperparameters
- We should strive for more transparent and realistic experimental setups
  - Modern tooling (`wandb`) simplifies reporting results under various considerations

Thank You For Your Attention!

Further Results

Benefits Task-Dependent (-Agnostic) for ZS-XLT (FS-XLT)



- Weight Averaging Consistently On Par Or Better Than Baselines
- Magnitude of Benefits Depend on No. of Shots and Task
- Run-Averaging Curriculum Outperforms Hyperparameter Tuning

Model Averaging Makes ZS-XLT More Robust