NRC Systems for the WMT2025-LRSL Shared Task

Samuel Larkin **Chi-kiu Lo 羅致翹** **Rebecca Knowles**
Digital Technologies Research Centre
National Research Council Canada (NRC-CNRC)
{samuel.larkin,chikiu.lo,rebecca.knowles}@nrc-cnrc.gc.ca

Abstract

We describe the NRC team systems for the WMT25 Shared Tasks on Large Language Models (LLMs) with Limited Resources for Slavic Languages. We participate in the Lower Sorbian and Upper Sorbian Machine Translation and Question Answering tasks. On the machine translation tasks, our primary focus, our systems rank first according to the automatic MT evaluation metric (chrF). Our systems underperform on the QA tasks.

1 Introduction

This paper describes our systems submitted to the WMT 2025 LLMs with Limited Resources for Slavic Languages Shared Task. We focused primarily on LLM-based machine translation (MT) into Upper Sorbian (hsb) and Lower Sorbian (dsb). Balancing two competing tasks typically comes at the cost of performance on one task or the other. While we performed preliminary experiments on QA, our initial results were unsatisfactory and we chose to submit a system that was trained only for MT as our primary submission.

2 Data and Models

We constrained our systems to the corpora offered by the organizers. The WMT25 corpora are available in the shared task github repository[1] while the previous years' corpora are available in the WMT22 repository.[2] MT data is described in more detail in Section 3.1 while QA data is described in more detail in Section 4.1.

As required by the shared task, we trained our models using the 0.5B, 1.5B, and 3B size Qwen2.5 models as the base. Our submitted system for the shared task is based on `Qwen2.5-1.5B-Instruct`.[3]

3 Machine Translation

We focused primarily on MT performance in our submissions. In this section, we describe preliminary experiments on MT, leading to the choices we made for our final submissions (Section 5). We used the `doc_to_text` task prompt suggested in the `lm-eval` test harness:[4]

```
Translate the following German
text to Lower Sorbian.  Put it in
this format <dsb> Lower Sorbian
translation </dsb>.\n<deu> {{de}}
</deu>
```

as our machine translation prompt for Lower Sorbian (replacing `Lower Sorbian` by `Upper Sorbian` and `dsb` by `hsb` in the Upper Sorbian setting). In addition to exploring hyperparameters for supervised finetuning using `LLaMa-Factory`, we tested which combinations of language pairs and directions to use for training as well as different stopping criteria and prompt templates.

3.1 Data

We used almost all corpora for training except the *dev* from 2025 (see Table 1 for details).[5] We used *2025_dev* to evaluate translation quality. To avoid data contamination in our experimental evaluations

[1] https://github.com/TUM-NLP/llms-limited-resources2025

[2] https://github.com/mariondimarco/WMT22_UnsupVeryLowResMT_Data

[3] https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct

[4] https://github.com/TUM-NLP/wmt25-lrsl-evaluation/blob/main/lm_eval/tasks/wmt25-lrsl/sorbian/deu-dsb/deu-dsb.yaml

[5] *2022.train_dsb_hsb_62564.dsb-hsb*, which contains 62,565 sentence pairs with 673,781 Lower Sorbian words and 654,445 Upper Sorbian words was unintentionally omitted from training. This corpus is equivalent in size (in lines) to roughly 7% of the raw data we used for training, roughly 26% the size (in words) of the Lower Sorbian data we used for training, and roughly 8% the size (in words) of the Upper Sorbian data we used in training. Due to time constraints, we were unable to perform experiments to determine the extent to which including this might have changed model performance.

Proceedings of the Tenth Conference on Machine Translation (WMT), Volume 2: Shared Task Papers, pages 1133–1142
November 8–9, 2025. ©2025 His Majesty the King in Right of Canada, as represented by the National Research Council, Canada

on the *dev* data, we remove the 2025 *dev* set from all the training material using sentence pairs as the duplication key. Table 1 shows the number of sentences and words for each corpora before and after filtering. We sampled 200 sentences from the 2025 *dev* sets to produce *2025.dev_sample.de-dsb* and *2025.dev_sample.de-dsb* respectively. The smaller samples were used for evaluation during training to track training progress (without the cost of decoding the full *dev* set), while the full *dev* sets were used for our internal evaluations and model choices.

We preprocessed all MT corpora by removing control characters, removing carriage return (`\x0D`). We also collapsed tabs to space, folded multiple spaces into a single space and removed trailing spaces. See Appendix A for details.

3.2 Training

Training for MT was performed using `LLaMa-Factory`'s implementation of supervised finetuning (called the `sft stage`). We took inspiration from GemmaX2 (Cui et al., 2025) for our base hyperparameters. They provide their configuration in their github repository.[6] Table 2 lists the hyperparameters that we explored during training for MT; we did not perform a full grid search of all parameters. The specific values used in our final submission are given in Section 5. All experiments were performed on Tesla V100-SXM2-32GB.

3.3 Language Pairs and Translation Direction

In low-resource machine translation research, it has often been considered beneficial to train on as much language data as possible, sometimes incorporating training data from related languages. This has been a component of past shared tasks on Sorbian languages (Fraser, 2020; Libovický and Fraser, 2021; Weller-Di Marco and Fraser, 2022). We wanted to explore this in the LLM setting: should we finetune LLMs separately for each target language's corpora or combine Upper Sorbian and Lower Sorbian corpora to train a single LLM that could translate into both languages?[7] We were also interested in determining how unidirectional training (training only to translate into Upper Sorbian and/or Lower Sorbian) compared to bidirectional training (training to translate into Upper Sorbian and/or Lower Sorbian as well as into German). Table 3 and Table 4 show that according to chrF,[8] the best-scoring combination of corpora and direction was a single system that incorporates both language pairs in both directions.[9] However, the chrF difference between the unidirectional training and the bidirectional training was consistently small or non-existent. Given the computational cost of training bidirectionally (twice as much data), we opted to use a unidirectional setup.

While running early experiments, we briefly considered LoRA (Hu et al., 2021) training, but observed substantially lower performance in the range of 12-15 BLEU and 7-10 chrF. For this reason, we continued to train full model weights in the remainder of our experiments.

3.4 Stopping Criterion

To improve training time, we can take advantage of the early stopping criterion which halts training if a given criterion or metric does not improve for a given number of gradient updates known as steps. Our baseline consists of training for exactly 5 epochs (`#epochs`), which we compared with a cross-entropy loss (`eval_loss`) approach where the model stops when the evaluation loss does not improve for 10 steps, and a `chrF` approach that stops when chrF has not improved for 10 steps. The models for these experiments were unidirectional (into Sorbian) translation models trained on the German to Upper or Lower Sorbian data as well as the Upper Sorbian–Lower Sorbian data.

Table 5 and Table 6 show that training for a fixed number of epochs always outperformed the other stopping criteria, yielding higher chrF scores. We used a fixed number of epochs for our final submitted system.

3.5 LLM Template

`LLaMa-Factory` offers multiple templates to format the input examples and prompt. We tested three of these built-in templates for the translation task: `qwen` (since the model we train is Qwen-

[6] https://github.com/xiaomi-research/gemmax/blob/main/scripts/sft.sh

[7] We included Lower Sorbian–Upper Sorbian parallel data in these experiments in addition to the German corpora: *2022.dev_dsb_hsb_new.dsb-hsb* and *2022.valid_dsb_hsb.dsb-hsb* are Lower Sorbian–Upper Sorbian corpora and because of that, during training, they were seen in both directions—effectively doubling their sentence count contribution.

[8] `nrefs:1|case:mixed|eff:yes|nc:6|nw:0|space:no|version:2.5.1`

[9] For completeness, we show Upper Sorbian systems' translations of Lower Sorbian text as well as the reverse, but show them in parentheses.

	raw				cleaned			
corpora	sentence	de_word	dsb_word	hsb_word	sentence	de_word	dsb_word	hsb_word
2020.devel.hsb-de.de-hsb	2000	24413		21692	1986	24246		21540
2020.devel_test.hsb-de.de-hsb	2000	24482		22081	1981	24241		21865
2020.train.hsb-de.de-hsb	60000	724572		639740	59703	720816		636414
2021.devel.dsb-de.de-dsb	601	7722	7231		601	7722	7231	
2021.devel_test.dsb-de.de-dsb	602	7786	7239		602	7786	7239	
2021.train.hsb-de.de-hsb	87521	1251339		1094421	86719	1240939		1085235
2022.40194_train_dsb_de.de-dsb	40194	514843	468509		40194	514843	468511	
2022.dev_dsb_hsb_new.dsb-hsb	700		7651	7416	700	700		
2022.HSB-DE_dev.tsv.de-hsb	2000	25607		22731	34	441		405
2022.HSB-DE_train.tsv.de-hsb	301536	3986351		3501610	301536	3986351		3501610
2022.valid.de-dsb	1353	9852	8424		1	2	3	
2022.valid_dsb_hsb.dsb-hsb	709		4592	4539	709	709		
2025.train.de-dsb	171964	2209255	2044017		171964	2209256	2044023	
2025.train.de-hsb	187270	2965088		2676309	187270	2965088		2676309
total	858450	11751310	2547663	7990539	854000	11703140	2527007	7943378
2025.dev.de-dsb	4000	46577	42295		4000	46577	42295	
2025.dev.de-hsb	4000	57580		51605	4000	57580		51605
2025.dev_sample.de-dsb					200	2291	2102	
2025.dev_sample.de-hsb					200	2855		2575

Table 1: Sentence count, per language word count of corpora. Training corpora are in the top section whereas the development sets are in the lower part of the table.

hyperparameter	values
per_device_train_batch_size	`4`, **`8`**, `32`, `64`
learning_rate	`1.0e-05`, `2.0e-05`, **`7.0e-05`**
num_train_epochs	`1`, `5`, `10`, **`20`**, `100`
gradient_accumulation_steps	`4`, **`8`**, `16`, `32`
max_grad_norm	**`1.0`**
warmup_ratio	**`0.0`**, `0.01`
weight_decay	**`0.0`**, `0.01`
lr_scheduler_type	**`cosine`**, `inverse_sqrt`
template	**`chatml`**, `empty`, `qwen`

Table 2: The values of hyperparameters that were investigated in various combinations, but not through a complete Cartesian product. Values in **bold** are from our submission.

languages	weights	unidir.	bidir.
de-dsb	full	70.0	70.5
de-{dsb,hsb}	full	75.7	75.8
de-{dsb,hsb}	LoRA	44.1	
de-hsb	full	(36.1)	(45.1)

Table 3: chrF scores for German to Lower Sorbian translations of *2025.dev.de-dsb* comparing unidirectional (into Sorbian {dsb,hsb}) vs. bidirectional (into Sorbian {dsb,hsb} and into German) corpora and different language pair combinations. These models are supervised finetuned (sft) from Qwen2.5-0.5B-Instruct. For completeness, in parentheses, we show Upper Sorbian systems' translations of Lower Sorbian.

languages	weights	unidir.	bidir.
de-dsb	full	(42.3)	(42.6)
de-{dsb,hsb}	full	80.2	80.2
de-{dsb,hsb}	LoRA	54.8	
de-hsb	full	79.2	79.2

Table 4: chrF scores for German to Upper Sorbian translations of *2025.dev.de-hsb* comparing unidirectional (into Sorbian {dsb,hsb}) vs. bidirectional (into Sorbian {dsb,hsb} and into German) corpora and different language pairs combinations. These models are supervised finetuned (sft) from Qwen2.5-0.5B-Instruct. For completeness, in parentheses, we show Lower Sorbian systems' translations of Upper Sorbian.

	Stopping Criterion		
model size	`eval_loss`	`#epochs`	`chrF`
0.5B	75.1	75.7	69.8
1.5B	76.7	77.7	74.7
3B	79.1	79.8	75.5

Table 5: chrF↑ scores for German to Lower Sorbian translations of *2025.dev.de-dsb* using different stopping criteria.

	Stopping Criterion		
model size	`eval_loss`	`#epochs`	`chrF`
0.5B	80.0	80.2	77.0
1.5B	81.8	82.2	80.9
3B	83.0	83.5	81.6

Table 6: chrF↑ scores for German to Upper Sorbian translations of *2025.dev.de-hsb* using different stopping criteria.

chrF↑	template		
Languages	chatml	empty	qwen
de-dsb	70.1	70.0	70.0
de-{dsb,hsb}	75.7	75.7	75.6
de-hsb	(40.6)	(36.1)	(40.7)

Table 7: chrF↑ scores for German to Lower Sorbian on *2025.dev.de-dsb* using different templates. These models are supervised finetuned(sft) from Qwen2.5-0.5B-Instruct. For completeness, in parentheses, we show Upper Sorbian systems' translations of Lower Sorbian.

chrF↑	template		
Languages	chatml	empty	qwen
de-dsb	(42.4)	(42.3)	(42.4)
de-{dsb,hsb}	80.2	80.2	80.1
de-hsb	79.4	79.2	79.3

Table 8: chrF↑ scores for German to Upper Sorbian on *2025.dev.de-hsb* using different templates. These models are supervised finetuned(sft) from Qwen2.5-0.5B-Instruct. For completeness, in parentheses, we show Lower Sorbian systems' translations of Upper Sorbian.

based), `chatml` (a generic chat template), and `empty` (a non-chat template). Figure 1 shows example input for the `empty` template. Figure 2 shows example input for the `chatml` and `qwen` template. The `qwen` and `chatml` templates, given the same input, render the same output except that `qwen`'s output has an additional system prompt. The system prompt is: "*You are Qwen, created by Alibaba Cloud. You are a helpful assistant*". Figure 4 shows a rendered example, using `qwen`'s template, as seen at training time by `LLaMa-Factory`.

We trained 9 models, the Cartesian product of three templates with three language corpora, using the identical configurations except for the datasets and the template. The models were trained to translate from German into Upper Sorbian and/or Lower Sorbian. We then translated the *2025.dev.de-dsb* and *2025.dev.de-hsb* sets using `LLaMa-Factory`. Table 7 shows chrF scores for translations into Lower Sorbian and Table 8 shows chrF scores for translation into Upper Sorbian when training a system on a given set of languages and template.[10] Naturally, we would not expect that a model trained on de-dsb be particularly good at translating Upper Sorbian or a de-hsb model at translating Lower Sorbian. This is confirmed in Table 7 and Table 8. Otherwise, the template choice does not seem to substantially impact the translation quality. The earlier translation direction and corpora experiments as well as the stopping criterion experiments were performed using the `empty` template.

4 Question Answering

We used our MT-finetuned models as the base models from which to train for the QA task. Our main experimental method for Multiple Choice Question Answering (MCQA) training was to use Direct Preference Optimization (Rafailov et al., 2024). We also ran limited experiments using supervised finetuning, but initial results showed a large drop in performance on the translation task as a result.

4.1 Data

We used all the available QA data for Upper Sorbian and Lower Sorbian. In Section 4.1.1 we describe how we split the data into *dev* and *train*. Section 4.1.2 discusses mitigation for potential position bias and Section 4.1.3 describes an approach to augmenting the data to try to better handle the range of possible response list sizes.

4.1.1 Dev/Train Split

To track QA performance during training, we split the QA corpora provided by the organizers[11] into 20% *dev* set and 80% *train*. Sampling of *dev* was performed using an unweighted reservoir sampling[12] algorithm. Table 11 shows which document ids were used in our *dev* set.

4.1.2 Position Bias

Position bias, where the position of answers influences the model's accuracy in answering, is known to be a problem for LLMs (Pezeshkpour and Hruschka, 2024, i.a.) and this problem can be exacerbated if there is also bias in the training data (i.e., if the correct answer is frequently shown in the same position among the options). To prevent answer position bias, we did data augmentation by adding repeated versions of the questions after permuting the answer order. We generate up to a maximum of 16 permutations per question-answer pair. That is,

[10] For completeness, we show Upper Sorbian systems' translations of Lower Sorbian text as well as the reverse, but show them in parentheses.

[11] https://github.com/TUM-NLP/llms-limited-resources2025

[12] https://en.wikipedia.org/wiki/Reservoir_sampling

```
{
  "instruction": "Translate the following German text to Lower
  ↪  Sorbian. Put it in this format <dsb> Lower Sorbian translation
  ↪  </dsb>.",
  "input": "<deu> Sogar Franz, der überhaupt nicht gerne singt,
  ↪  brummt laut mit. </deu>",
  "output": "<dsb> Samo Franc, kenž zewšym rad njespiwa, barcy
  ↪  głosnje sobu. </dsb>"
}
```

Figure 1: Input example for `empty` template

```
{
  "conversations": [
    {
      "from": "human",
      "value": "Translate the following German text to Lower Sorbian.
       ↪  Put it in this format <dsb> Lower Sorbian translation
       ↪  </dsb>.\n<deu> Sogar Franz, der überhaupt nicht gerne
       ↪  singt, brummt laut mit. </deu>"
    },
    {
      "from": "gpt",
      "value": "<dsb> Samo Franc, kenž zewšym rad njespiwa, barcy
       ↪  głosnje sobu. </dsb>"
    }
  ]
}
```

Figure 2: Input example for `chatml`/`qwen` template

we generate the minimum between the factorial of the number of possible answer orders for a given question or 16.

4.1.3 Augmentation

Noting that `lm-eval` considers all 16 possible answer positions and that many of the questions did not have a total of 16 answers, we augmented the answer sets during training such that each augmented QA instance had the correct answer paired with an incorrect answer in one of the other 15 answer positions. This was done after the position bias augmentation. Direct preference optimization then contrasts the correct and rejected answer during training. Table 9 shows the final number of questions in each QA corpora. Doing this augmentation after the position bias augmentation may have undermined the effectiveness of that position bias mitigation approach, as this second augmentation may reinforce a bias towards the first few answers being the correct response.

	original		augmented	
Question	*dev*	*train*	*dev*	*train*
DSB_A1	6	24	180	720
DSB_A2	5	23	210	1350
DSB_B1	8	36	1215	5250
DSB_B2	11	45	1695	7830
HSB_A1	6	24	180	720
HSB_A2	5	23	210	1350
HSB_B1	8	36	1215	5250
HSB_B2	11	45	1695	7830

Table 9: Number of questions per question type.

4.2 Training

To train our models for the QA task, we once again used `LLaMa-Factory` to tune all weights but used direct preference optimization (called the

	chrF↑		accuracy	
	dsb	hsb	dsb-qa	hsb-qa
TartuNLP	78.20	86.33	**57.56**	**58.10**
NRC	**78.24**	**87.20**	32.20	29.05
SDKM	64.34	75.73	51.71	55.24
baseline	12.21	13.88	45.85	42.86

Table 10: Official results.

`dpo stage` in `LLaMa-Factory`) rather than supervised finetuning. We explored learning rates of `7.0e-05`, `2.0e-05`, `7.0e-06`, `2.0e-06`, `2.0e-07` and `1`, `3` or `5` training epochs. The gradient norm was set to `1.0`, the warmup ratio to `0.01`, weight decay to `0.01`. The learning rate scheduler was set to `inverse_sqrt` and the optimizer was `adamw_torch`. The effective batch size was `32` because we used a per device batch size of `1` and `8` gradient accumulation steps over `4` GPUs. We encountered challenges in training a 3B parameter model for QA, primarily due to insufficient GPU memory to accommodate even a batch size of one. Consequently, we were restricted to smaller model sizes, specifically the 0.5B and 1.5B parameter variants. The inability to train a 3B parameter model for QA, coupled with the competition's requirement to employ a singular model for both MT and QA, rendered any further attempts to train a 3B parameter model for MT futile. Figure 6 and Figure 7 illustrate a correct answer and a rejected answer respectively, rendered from the corpus sample in Figure 5 the QA template.

4.3 Results

We evaluated on our QA *dev* sets and saw improvements over random chance. However, later examination of lower-than-expected scores when evaluating on the training data indicated that the models are not training as well we would have hoped. Table 14 shows our *dev* and *train* accuracies of our submission system compared to random chance, highlighting that our system was not effectively learning.

We are continuing to explore the reasons behind this, whether it relates to the data augmentation approach, the data splits for training, the fact that our *dev* was quite small, the selection of hyperparameters, the choice to use DPO, or some combination thereof.

5 Submissions

Here, we describe the specifics of our primary submissions. We submitted a single model that was trained to translate into both Upper Sorbian and Lower Sorbian. Note that our translations were obtained from `LLaMa-Factory` for inference while the QA output are produced using `lm-eval`.[13]

At submission time our best model attempting to balance between MT and QA performance, according to our *dev* sets for MT and QA, was based on `Qwen2.5-1.5B-Instruct`[14]. It used `LLaMa-Factory`'s supervised finetuning (`sft training stage`) and trained all weights (not LoRA). Using the `chatml` template, it trained on our filtered corpora from the top part of Table 1: 855,409 training examples as reported in the log. The learning rate was `7e-5` with a learning rate scheduler of type `cosine`. We capped the maximum gradient norm of `1.0`, and used no warmup and no weight decay. We used `8` gradient accumulation steps, a per device training batch size of `8` on `4` GPUs for a total training batch size of `256`. As this was a model that we trained near the start of the task, not knowing what would be a good number of training epochs given the number of training examples, we arbitrarily used `20` epochs (66,820 updates) and left the model to train until completion, expecting that the training loss curve would guide us in selecting a smaller number of epochs for future training sessions. Over the entire training, this model has seen 2,941,371,584 input tokens and required 2.31e+19 Flops of computation.

The official results for the task are shown in Table 10. As might be expected for MT-only systems, we have the highest performance on the MT tasks, but the lowest performance on QA, even falling below random chance for HSB-QA. This was in contrast to the performance that we observed on development data (where we did improve on random chance, even for MT-only systems), which merits additional analysis. Our leading hypotheses are that our *dev* set was too small and that DPO was not suited for the QA task.

We note that, after the late decision to use an MT-only system, we might have benefited from

[13] Our removal of newlines for the MT training data meant that we did not typically meet the expected stopping criteria for `lm-eval`, hence our use of `LLaMa-Factory` for inference.

[14] https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct

selecting a 3B parameter model rather than a 1.5B parameter model, as we did observe increasing MT performance with increasing model size in early stopping criterion experiments. While our stopping criterion experiments showed that chrF scores increased as the number of model parameters increased, with 3B outperforming 1.5B, we had paused our work on 3B parameter models in order to focus on the QA experiments. Had we run additional MT training (in line with our final setup the submission) on 3B model parameters, we expect they would have outperformed our submitted 1.5B parameter model results. We opted not to submit an early 3B MT-only model.

6 Conclusion

We submitted systems for the Upper Sorbian and Lower Sorbian portions of the shared task. Due to the lack of success of our approaches to QA, we submitted systems that were trained only on MT. As expected, our systems performed well on translation (our main focus) but underperformed on QA. If we had examined the QA accuracies on our training data earlier, as shown in Table 14, we could have concluded that our training for QA was ineffective. We were misled by the performance on the development set, which may have been too small for the task; larger development sets or cross-validation approaches could have also alerted us to these issues earlier. For MT for these low-resource languages, we observed benefits of training using both Lower Sorbian and Upper Sorbian parallel text (on the order of 5 chrF points for Lower Sorbian and 1 chrF point for Upper Sorbian).

We had observed some small improvements in QA even when training only on MT when evaluated on *dev* data, but did not find the same results on the test set. We continue to explore the reasons for this.

Limitations

We focused primarily on the machine translation portion of the task. There remain areas to explore in more detail, such as the impacts of model size (number of parameters) on performance. For some of our experiments, such as comparing unidirection and bidirectional training data, we trained the smallest (0.5B parameter) models; it is not known how well these results will generalize as the number of model parameters scales. There remains additional work to do on the QA task and on analyzing why our training for that did not perform as expected.

Acknowledgements

We would like to thank the reviewers for their suggestions and comments and the shared task organizers for their work on the task.

References

Menglong Cui, Pengzhi Gao, Wei Liu, Jian Luan, and Bin Wang. 2025. Multilingual machine translation with open large language models at practical scale: An empirical study.

Alexander Fraser. 2020. Findings of the WMT 2020 shared tasks in unsupervised MT and very low resource supervised MT. In *Proceedings of the Fifth Conference on Machine Translation*, pages 765–771, Online. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Jindřich Libovický and Alexander Fraser. 2021. Findings of the WMT 2021 shared tasks in unsupervised MT and very low resource supervised MT. In *Proceedings of the Sixth Conference on Machine Translation*, pages 726–732, Online. Association for Computational Linguistics.

Pouya Pezeshkpour and Estevam Hruschka. 2024. Large language models sensitivity to the order of options in multiple-choice questions. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2006–2017, Mexico City, Mexico. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model.

Marion Weller-Di Marco and Alexander Fraser. 2022. Findings of the WMT 2022 shared tasks in unsupervised MT and very low resource supervised MT. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 801–805, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

A Data Cleaning

Figure 3 shows the regular expression used to perform clean up of the corpora.

B Dev Question IDs

Table 11 lists the QA question ids for items in the *dev* set.

Question	ids
DSB_A1	A1.1.H02, A1.1.H11, A1.1.H7, A1.1.L14, A1.1.L6, A1.1.L7
DSB_A2	A2.1.H02, A2.1.H11, A2.1.L12, A2.1.L14, A2.1.L3
DSB_B1	B1.1.S12, B1.1.S13, B1.1.S19, B1.1.S23, B1.1.S3, B1.1.S5, B1.1.S7, B1.1.S9
DSB_B2	B2.1.H17, B2.1.H3, B2.1.H5, B2.1.L14, B2.1.L8, B2.1.S18, B2.1.S19, B2.1.S22, B2.1.S23, B2.1.S7, B2.1.S8
HSB_A1	A1.1.H02, A1.1.H11, A1.1.H7, A1.1.L14, A1.1.L6, A1.1.L7
HSB_A2	A2.1.H02, A2.1.H11, A2.1.L12, A2.1.L14, A2.1.L3
HSB_B1	B1.1.S12, B1.1.S13, B1.1.S19, B1.1.S23, B1.1.S3, B1.1.S5, B1.1.S7, B1.1.S9
HSB_B2	B2.1.H17, B2.1.H3, B2.1.H5, B2.1.L14, B2.1.L8, B2.1.S18, B2.1.S19, B2.1.S22, B2.1.S23, B2.1.S7, B2.1.S8

Table 11: Questions ids for *dev* sets.

```
s/[\x01-\x09\x0B\x0C\x0E-\x1D\x7F]//g;
s/\x0D//g;
s/\t/ /g;
s|\\\\ ?[rn]| |g;
s|\\ ?[rn]| |g;
s/  */ /g;
s/  *$//;
```

Figure 3: Regular expressions used to clean up input corpora.

C Random Baselines

Table 12 lists the random chance baseline for selecting correct responses on the *dev* set, while Table 13 shows the same for the test set.

Question	DSB(%)	HSB(%)
A1	50.00	50.00
A2	42.86	42.86
B1	25.19	25.19
B2	21.74	21.74
Overall	31.81	31.81

Table 12: *dev* set random chance baseline accuracy.

Question	DSB(%)	HSB(%)
A1	50.00	50.00
A2	42.86	42.86
B1	25.19	25.19
B2	21.74	21.74
C1	32.71	31.15
Overall	31.93	31.65

Table 13: Test set random chance baseline accuracy.

D Supervised Finetuning

Figure 4 shows Figure 2 rendered using Qwen's template.

E Question Answering

Figure 5 illustrates a training example for QA. Figure 6 is the resulting output of a correct answer whereas Figure 7 is the resulting output of a rejected answer as seen at training time by the model.

F Submission QA Accuracies

Table 14 shows random chance accuracies alongside the accuracies of our submitted system on both *dev* and *train*, emphasizing the inadequate training.

Source

```
<|im_start|>system
You are Qwen, created by Alibaba Cloud. You are a helpful
 ↪  assistant.<|im_end|>
<|im_start|>user
Translate the following German text to Lower Sorbian. Put it in this
 ↪  format <dsb> Lower Sorbian translation </dsb>.
<deu> Sogar Franz, der überhaupt nicht gerne singt, brummt laut mit.
 ↪  </deu><|im_end|>
<|im_start|>assistant
<dsb> Samo Franc, kenž zewšym rad njespiwa, barcy głosnje sobu.
 ↪  </dsb><|im_end|>
```

Target

```
<dsb> Samo Franc, kenž zewšym rad njespiwa, barcy głosnje sobu.
 ↪  </dsb><|im_end|>
```

Figure 4: Rendered `qwen` template for MT.

```
{
  "instruction": "žona: Pětš, psiźoš sobu do kina? \nPětš: Halo,
   ↪  Monika. Njewěm hyšći. Ga? \nžona: Pónjeźele? \nPětš: Derje,
   ↪  pónjeźele.\n\nQuestion:\nGa cotej Pětš a Monika do kina
   ↪  hyś?\n\nPossible answers:\n1 pónjeźele\n2 pětk\n\nAnswer:",
  "chosen": "1",
  "rejected": "2"
}
```

Figure 5: Input example for Question Answering using DPO template

```
žona: Pětš, psiźoš sobu do kina?
Pětš: Halo, Monika. Njewěm hyšći. Ga?
žona: Pónjeźele?
Pětš: Derje, pónjeźele.

Question:
Ga cotej Pětš a Monika do kina hyś?

Possible answers:
1 pónjeźele
2 pětk

Answer:1
```

Figure 6: Rendered QA template for the chosen answer

```
žona: Pětš, psiźoš sobu do kina?
Pětš: Halo, Monika. Njewěm hyšći. Ga?
žona: Pónjeźele?
Pětš: Derje, pónjeźele.

Question:
Ga cotej Pětš a Monika do kina hyś?

Possible answers:
1 pónjeźele
2 pětk

Answer:2
```

Figure 7: Rendered QA template for the rejected answer

	DSB					HSB				
Submission	Sorbian	A1	A2	B1	B2	Sorbian	A1	A2	B1	B2
dev	45.13	66.67	40.00	37.50	36.36	40.59	66.67	40.00	37.50	18.18
train	26.87	45,83	26.09	22.22	13.33	32.84	50.00	39.13	22.22	20.00
Random Chance	31.81	50.00	42.86	25.19	21.74	31.81	50.00	42.86	25.19	21.74

Table 14: Question Answering *dev*/*train* accuracies (%) for our submission.