Fine-Tuning Medium-Scale LLMs for Joint Intent Classification and Slot Filling: A Data-Efficient and Cost-Effective Solution for SMEs

Maia Aguirre[1,2], **Ariane Méndez**[1], **Arantza del Pozo**[1], **María Inés Torres**[2] and **Manuel Torralbo**[1]
[1] Vicomtech Foundation, Basque Research and Technology Alliance (BRTA)
[2] University of the Basque Country (UPV/EHU)
{magirre,amendez,adelpozo,mtorres,mtorralbo}@vicomtech.org

Abstract

Dialogue Systems (DS) are increasingly in demand for automating tasks through natural language interactions. However, the core techniques for user comprehension in DS depend heavily on large amounts of labeled data, limiting their applicability in data-scarce environments common to many companies. This paper identifies best practices for data-efficient development and cost-effective deployment of DS in real-world application scenarios. We evaluate whether fine-tuning a medium-sized Large Language Model (LLM) for joint Intent Classification (IC) and Slot Filling (SF), with moderate hardware resource requirements still affordable by SMEs, can achieve competitive performance using less data compared to current state-of-the-art models. Experiments on the Spanish and English portions of the MASSIVE corpus demonstrate that the Llama-3-8B-Instruct model fine-tuned with only 10% of the data outperforms the JointBERT architecture and GPT-4o in a zero-shot prompting setup in monolingual settings. In cross-lingual scenarios, Llama-3-8B-Instruct drastically outperforms multilingual JointBERT demonstrating a vastly superior performance when fine-tuned in a language and evaluated in the other.

1 Introduction

Dialogue Systems (DS) are experiencing unprecedented demand from companies and administrations, driven by their transformative ability to automate tasks through natural language communication (Altarif and Al Mubarak, 2022).

Intent Classification (IC) and Slot Filling (SF) are fundamental tasks for user comprehension in DS. IC identifies the user's communicative purpose and SF extracts essential details from their input. Nonetheless, these Natural Language Understanding (NLU) tasks are particularly challenging and often create bottlenecks in DS performance.

Current state-of-the-art techniques for joint IC and SF are based on the BERT (Devlin et al., 2019) model architecture and rely heavily on extensive labeled data (Zailan et al., 2023). However, industrial environments often lack such datasets, and the data annotation process for NLU is labor-intensive and requires expert annotators. This limitation restricts the widespread deployment of customized assistants across many companies (Aguirre et al., 2023).

Over the past few years, Large Language Models (LLM) have experienced a notable surge, significantly advancing the state-of-the-art in generative tasks such as Natural Language Generation (Minaee et al., 2024). Most LLM families release models in various sizes: ultra-large-scale models (over 70 billion parameters) excel in complex tasks with high accuracy; large-scale models (over 13 billion parameters) balance performance and resource needs; medium-scale models (over 7 billion parameters) are efficient with moderate resources. Recently, smaller language models have emerged targeting edge devices, offering reasonable performance and resource efficiency for real-time applications (Mehta et al., 2024).

Various studies have explored the use of ultra-large scale LLMs for NLU tasks employing zero-shot and few-shot techniques, achieving highly promising results with little data requirements. However, these LLMs demand substantial resources for deployment and are typically accessed through online services hosted in the cloud by multinational corporations. Moreover, companies and small and medium-sized enterprises (SMEs) that prioritize data privacy and wish to avoid high long-term costs often favor models that can be deployed on-premise (Fortuna et al., 2023).

Therefore, this paper focuses on identifying best practices and practical considerations for data-efficient development and cost-effective deployment of NLU models in real-world application scenarios. To achieve this, it explores whether medium-scale LLMs, which demand only mod-

Proceedings of the 31st International Conference on Computational Linguistics: Industry Track, pages 251–262
January 19–24, 2025.

erate hardware resources for deployment, can enhance the joint IC and SF tasks with less annotated data, while still delivering competitive performance compared to the current leading models. The study systematically compares the performance of a BERT-based architecture for joint IC and SF with that of a fine-tuned medium-scale LLM, varying the amount of training data incrementally to assess their efficiency.

The analysis considers two scenarios: monolingual and cross-lingual. In the monolingual setting, models are fine-tuned and evaluated separately for each language to compare performance across linguistic settings. In the cross-lingual scenario, models are fine-tuned in one language and evaluated in another to assess their ability to transfer learning across languages. This cross-lingual capability shall help further reduce the amount of annotated data needed for the development of practical multilingual DS, enabling companies to lower data annotation costs while maintaining strong NLU performance.

This study is the first to show that, in monolingual settings, fine-tuning a mid-scale LLM for joint IC and SF with just 10% of the data outperforms the traditional BERT-based architecture trained on the full dataset and also exceeds the performance of zero-shot GPT-4o. Furthermore, cross-lingual experiments confirm the medium-scale model's exceptional language transfer capabilities compared to the BERT-based architecture.

The rest of the paper is organized as follows: Section 2 reviews recent work on IC and SF tasks using LLMs. Section 3 describes the main characteristics of the corpus used and explains how the dataset was sampled to experiment with varying amounts of training data. Section 4 presents the specifications of the models employed. Section 5 presents the results obtained from the various experiments conducted, and finally, Section 6 summarizes the main conclusions and suggests possible directions for future work.

2 Related Work

Several studies using pretrained language models such as BERT (Devlin et al., 2019) have demonstrated that jointly addressing IC and SF tasks enhances performance on both, highlighting their strong correlations (Weld et al., 2022). However, despite these advances, the effectiveness of these models still relies heavily on the availability of extensive labeled data (Hu et al., 2023).

While LLMs hold promise in potentially reducing the necessity for extensive labeled data, their complete capabilities for NLU have only recently begun to be explored.

Recent studies have primarily concentrated on investigating the potential of LLMs for NLU tasks utilizing zero-shot and few-shot prompting, demostrating that larger models generally outperform smaller ones, especially in IC with a limited number of intents. However, their performance declines as the number of intents increases as well as in SF tasks, and they have yet to surpass the current state-of-the-art. Parikh et al. (2023) state that instruction fine-tuned language models such as Flan-t5-xxl and GPT-3 are very effective in zero-shot settings with in-context prompting for IC, although they do not outperform state-of-the-art models. Additionally, He and Garner (2023) assess ChatGPT and OPT models of various sizes across multiple benchmarks. They observe that these models in zero or few-shot settings achieve IC accuracy comparable to fine-tuned BERT models in diverse languages, but encounter challenges with SF tasks. Furthermore, they note that smaller models, such as OPT-6.7B, demonstrate significantly poorer performance. A further study using ChatGPT in zero and few-shot settings shows it excels with tasks involving few intents but struggles with those involving many (Wang et al., 2024). Concerning joint IC and SF, GPT-SLU (Zhu et al., 2024) introduces a two-step zero-shot prompt technique: initially extracting intents and slots separately, then alternating to extract intents given entities and vice versa. Although it demonstrates promising results, the evaluation is limited to ChatGPT and it does not exceed state-of-the-art performance.

A few very recent works have explored the fine-tuning of medium-scale LLMs for NLU tasks. Overall, fine-tuned medium-scale models have shown to outperform ultra-large LLMs in zero-shot and few-shot scenarios, achieving performance levels comparable to JointBERT using the same amount of training data. In the case of IC, multiple intent classification is explored in (Yin et al., 2024) by leveraging the Vicuna-7B-v1.5, Llama-2-7B-chat, and Mistral-7B-Instruct-v0.1 models. This approach involves defining sub-intents to identify which parts of the utterance relate to each intent. The study illustrates that in most cases, these LLMs marginally enhance the existing state-of-the-art. For the SF task, Shrivatsa Bhargav et al. (2024) fine-

tune Mistral-7B-Instruct-v0.1, Flan-t5-xl and a proprietary LLM granite.13b.v2 and demonstrate that using slot descriptions instead of the slot names enhances the model's performance, surpassing GPT-3.5 prompting. Lastly, regarding both IC and SF tasks, the study by (Mirza et al., 2024) evaluates the performance of the LoRA (Low-Rank Adaptation) fine-tuned Flan-t5-xxl model. LoRA is an efficient fine-tuning technique that reduces the number of trainable parameters by decomposing weight updates into low-rank matrices, significantly lowering memory and computational requirements while maintaining performance. The study trains the model separately for each task, integrating potential intent and slot candidates into the instruction prompts. As a result, the LoRA fine-tuned model outperforms GPT-3.5 in zero-shot and few-shot settings, and achieves close to the state-of-the-art performance of the JointBERT approach (Chen et al., 2019).

Previous studies, however, have not thoroughly examined the impact of training data quantity on fine-tuning LLM models for joint IC and SF tasks. To address this, our work investigates whether medium-scale LLMs can enhance data efficiency in this task compared to existing methods.

Additionally, LLMs are inherently multilingual, as they are pretrained on diverse multilingual corpora, demonstrating impressive reasoning abilities across a wide range of languages. The cross-lingual competencies of LLMs are being studied in different contexts (Chua et al., 2024; Hu et al., 2024). Some recent studies are also investigating cross-lingual IC/SF using state-of-the-art approaches based on multilingual BERT-like models, such as mBERT, XLM-R and mT5. These studies focus on efficient cross-lingual transfer learning techniques, including data augmentation methods like paraphrasing and machine translation (Kwon et al., 2023), as well as prompt-tuning strategies (Tu et al., 2024). Nevertheless, the zero-shot cross-lingual effectiveness of current joint IC and SF methods has not yet been thoroughly compared to that of fine-tuned medium-scale LLMs. To fill this gap, we also conduct zero-shot cross-lingual experiments, fine-tuning models only in English and evaluating them in Spanish, and vice versa.

In summary, most prior research on IC and SF tasks has focused on very large LLMs in zero-shot and few-shot settings with limited intent and entity variety. Fine-tuning medium-scale LLMs has generally resulted in performance comparable to JointBERT with the same amount of data. However, only one study has evaluated medium-scale LLMs for joint IC and SF, and even then, each task is trained separately. Our work addresses these gaps by jointly fine-tuning medium-scale LLMs for both IC and SF tasks, while also exploring the impact of varying training data quantities on model performance, providing new insights into data efficiency. Concerning cross-lingual IC/SF research, recent studies mainly focus on multilingual BERT-like models, while the zero-shot performance of fine-tuned medium-scale LLMs remains underexplored. To bridge this gap, we conduct zero-shot cross-lingual experiments between English and Spanish.

3 Data

The selected corpus for experimentation is MASSIVE (FitzGerald et al., 2023), chosen for its coverage across 18 domains and translation into 51 languages, including Spanish and English. It encompasses a total of 60 different intents and 55 slots. The sentence distribution is as follows: 11.514 for training, 2.033 for validation and 2.974 for testing.

From this point forward, all reported processing has been applied exclusively to the training dataset, which has been systematically sampled into subsets of different sizes for the experiments conducted. These subsets have been created maintaining the original proportions of the intents, aiming to observe the impact of reducing each label type by predetermined percentages.

Firstly, 302 duplicate sentences have been removed from the training corpus, reducing it to 11.212 sentences. Then, the corpus has been divided into 60 segments, each corresponding to a specific intent. Next, the first 5%, 10%, 20%, 30%, 50%, 75%, and 100% examples of each intent segment have been saved. Each partition has subsequently been merged into subsets containing different intents at the same percentage, resulting in seven partitions that preserve the original dataset's proportions. Table 5, provided in Appendix A, shows the number of examples that correspond to each intent for each partition.

4 Implementation Strategies

As in (FitzGerald et al., 2023), we have fine-tuned the publicly-available pretrained XLM-R model to perform joint intent classification and slot filling using the JointBERT architecture (Chen et al., 2019) as a baseline, leveraging the implementa-

tion found in https://github.com/monologg/JointBERT. This implementation takes advantage of a pretrained encoder with two distinct classification heads. The first head uses the encoder's pooled output to predict intents, while the second head uses the sequence output to predict slots (Chen et al., 2019). The training of the model is carried out with the Adam optimizer (Kingma and Ba, 2014), and the best-performing model checkpoint is selected based on the best overall exact match accuracy across the validation examples. Fine-tuning has been conducted using a single 12 GB Nvidia Titan X, with training completing in under 10 hours for the largest models. Detailed hyperparameters are provided in Appendix B.

On the other hand, Llama-3-8B-Instruct[1] has been fine-tuned for joint IC and SF with LoRA (Hu et al., 2022), initializing the base model with 8-bit precision. Each training sample includes a specific instruction to guide the model in classifying both intent and slots from the given input. The instruction includes the input sentence written either in Spanish or English, depending on the corpus used, and its corresponding intent and slots in the following format: *intent: <intent_name>, entities: <["entity_name1: entity_value1", "entity_name2: entity_value2"]>*. Intent and slot names are provided in English while slot values remain in the original language (Spanish or English), since they are substrings of the original sentence. The training of the model is carried out with the Adam-8-bit optimizer, and the best-performing model checkpoint is selected based on the lowest cross-entropy loss achieved on the validation dataset. Fine-tuning has been conducted using a single 48 GB Nvidia L40, with training completing in under 8 hours for the largest data partitions. Detailed hyperparameters and the complete prompt are listed in Appendix B.

Additionally, the Llama-3-8B-Instruct and GPT-4o models have also been tested in a zero-shot setting. The prompt used for utterance evaluation in these cases includes the entire list of intents and slots of the MASSIVE corpus. The complete prompt is as well provided in Appendix B.

Regarding computational requirements, Llama-3-8B-Instruct needs approximately 16GB of VRAM for LoRA fine-tuning and 8GB for inference with 8-bit precision[2], which can be deployed on a dedicated server with an upfront starting cost of $5000. In contrast, GPT-4o operates under a pricing model of $5 per million input tokens and $15 per million output tokens.

The most cost-effective option will depend on the model's usage. Assuming each input, including the zero-shot prompt and sentence, averages 600 tokens, while the labeled output averages 20 tokens, and factoring in a 5-year depreciation period for the server, on-premise solutions become more economical for workloads exceeding approximately 850 queries per day.

5 Results

5.1 Monolingual Setting

Using the intent partitions described in Section 3, both JointBERT and Llama3-8B-Instruct models have been fine-tuned separately in each language with different data percentages, resulting in the outcomes shown in Table 1.

The fine-tuned Llama-3-8B-Instruct model clearly outperforms the JointBERT model across data partitions, achieving superior results with just 10% of the data compared to JointBERT's performance with 100%, as highlighted in bold in Table 1. Figure 1 shows Intent Accuracy and Slot F1 metrics relative to the percentage of training data. While JointBERT more significantly improves with more data, Llama-3-8B-Instruct quickly attains higher accuracy and exhibits more marginal gains with further examples.

To gain a deeper understanding of intent classification performance, the Slot F1 score for each intent has been calculated across all partitions of the corpus. Table 2 provides a small excerpt of these results. This analysis indicates that the highest F1 scores do not necessarily align with the intents that have the most examples.

To better comprehend these results, Figure 2 presents a t-SNE (t-distributed Stochastic Neighbor Embedding) representation of the sentence embeddings, calculated using Sentence Transformers[3] with the uncased multilingual BERT model[4] of the intents in Table 2. The figure shows that intents with high F1 scores are closely clustered, while those with poor F1 scores, such as the "general_quirky" and "email_querycontact" classes, have scattered embeddings despite having many examples. This indicates that both the Llama-3-

[1]https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[2]https://huggingface.co/blog/llama31

[3]https://huggingface.co/sentence-transformers

[4]https://huggingface.co/google-bert/bert-base-multilingual-uncased

Model		JointBERT ES	JointBERT EN	Llama3 ES	Llama3 EN
5%	Int. acc	0.5218	0.6348	0.8073	0.8127
	Slot F1	0.2635	0.3423	0.5035	0.5111
	Exact m	0.2276	0.2781	0.4116	0.3679
10%	Int. acc	0.5965	0.6812	**0.8309**	**0.8527**
	Slot F1	0.3020	0.3886	**0.5757**	**0.6314**
	Exact m	0.2710	0.3241	**0.4926**	**0.5383**
20%	Int. acc	0.7192	0.7603	0.8453	0.8241
	Slot F1	0.3801	0.4605	0.6034	0.5774
	Exact m	0.3621	0.4126	0.5185	0.4842
30%	Int. acc	0.7492	0.7932	0.8581	0.8440
	Slot F1	0.4030	0.4963	0.6291	0.5950
	Exact m	0.3890	0.4512	0.5531	0.5084
50%	Int. acc	0.7787	0.8063	0.8682	0.8930
	Slot F1	0.4468	0.5717	0.6607	0.6989
	Exact m	0.4348	0.4929	0.5841	0.6193
75%	Int. acc	0.8016	0.8299	0.8722	0.8964
	Slot F1	0.4791	0.5844	0.6699	0.7010
	Exact m	0.4633	0.5378	0.5925	0.5965
100%	Int. acc	**0.8147**	**0.8376**	0.8796	0.8981
	Slot F1	**0.5075**	**0.6141**	0.6944	0.7646
	Exact m	**0.4916**	**0.5656**	0.6187	0.6856

Table 1: Intent Accuracy, Slot F1 and Exact match results for the JointBERT and Llama-3-8B-Instruct models fine-tuned on varying intent partitions of the Spanish and English MASSIVE corpus. The results in bold highlight the instance where Llama-3-8B-Instruct outperforms JointBERT trained with the entire dataset.

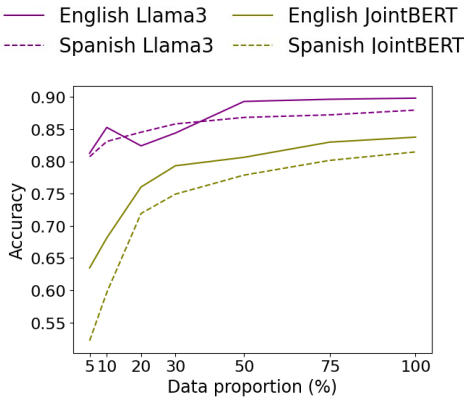
	#	JointBERT ES	JointBERT EN	Llama3 ES	Llama3 EN
general_quirky	546	0.5263	0.4841	0.6081	0.6768
email_query	411	0.9277	0.9483	0.9617	0.9614
calendar_remove	299	0.9489	0.9220	0.9412	0.9429
qa_currency	142	0.9268	0.9487	0.9744	0.9873
email_querycontact	127	0.6885	0.7273	0.8070	0.8235
transport_ticket	126	0.9268	0.9254	0.8889	0.9394
iot_cleaning	84	0.9412	0.9412	0.9811	0.9804
iot_wemo_off	38	0.8750	0.8889	0.9091	0.947

Table 2: Slot F1 Score of 8 example Intents for the JointBERT and Llama-3-8B-Instruct models fine-tuned with the 100% of the corpus. # represents the number of examples of that particular intent included in the fine-tuning.

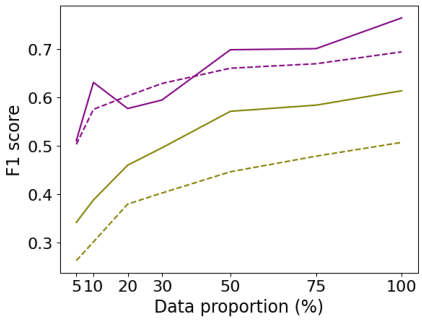
8B-Instruct and JointBERT models perform better when embeddings are either tightly clustered or well-separated from those of other intents, highlighting the importance of clear cluster boundaries over the sheer number of examples.

5.2 Cross-lingual Setting

To evaluate whether a medium-sized LLM can accurately detect intents and slots in a cross-lingual

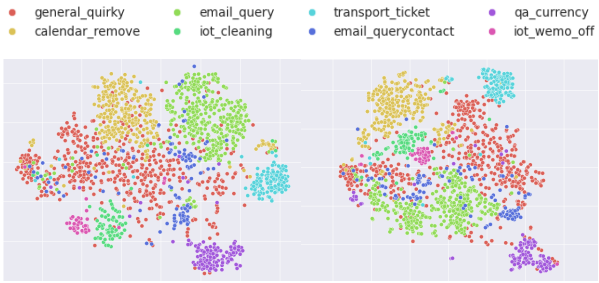


(a) Intent Accuracy



(b) Slot F1

Figure 1: Performance of Intent Classification Accuracy (a) and Slot Filling F1 score (b) for the Llama3 (purple) and JointBERT (green) models fine-tuned on different intent sample percentages of the English (solid) and Spanish (dashed) MASSIVE corpus.



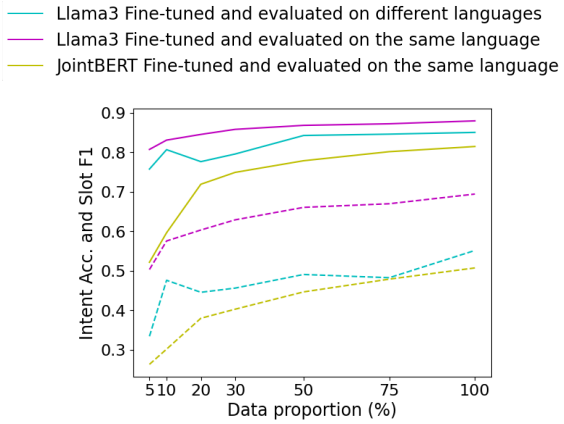
(a) Spanish (b) English

Figure 2: t-SNE visualization of sentence embeddings in the training set of the MASSIVE Spanish dataset, computed using sentence transformers with the uncased-multilingual-BERT model.

setting without access to domain-specific data in the target language, models fine-tuned exclusively in one language have been tested on the other language's test set. The results presented in Table 3 reveal a clear contrast in zero-shot cross-lingual performance between the XLM-R-based JointBERT architecture and Llama3-8B-Instruct, despite both leveraging multilingual pretraining.

Model		EN–>ES JB	EN–>ES Llama3	ES–>EN JB	ES–>EN Llama3
5%	Int. acc	0.1133	**0.7576**	0.1137	**0.8282**
	Slot F1	0.0325	**0.3345**	0.0404	**0.4700**
	Exact m	0.0219	**0.2377**	0.0343	**0.4021**
10%	Int. acc	0.1184	0.8067	0.1274	0.8531
	Slot F1	0.0313	0.4762	0.0445	0.5224
	Exact m	0.0235	0.4153	0.0407	0.4916
20%	Int. acc	0.1311	0.7764	0.2051	0.8625
	Slot F1	0.0398	0.4458	0.0648	0.5262
	Exact m	0.0269	0.3820	0.0528	0.5057
30%	Int. acc	0.1419	0.7957	0.01940	0.8739
	Slot F1	0.0430	0.4564	0.0701	0.5533
	Exact m	0.0252	0.4062	0.0514	0.5205
50%	Int. acc	0.1469	0.8426	0.2219	0.8847
	Slot F1	0.0419	0.4909	0.0731	0.5721
	Exact m	0.0303	0.4597	0.0548	0.5474
75%	Int. acc	0.1453	0.8460	0.2374	0.8837
	Slot F1	0.0460	0.4827	0.0708	0.5762
	Exact m	0.0252	0.4069	0.0525	0.5491
100%	Int. acc	**0.1496**	0.8504	**0.2384**	0.8897
	Slot F1	**0.0440**	0.5516	**0.0679**	0.5696
	Exact m	**0.0232**	0.5114	**0.0548**	0.5575

Table 3: Intent Accuracy, Slot F1 and Exact match results for the JointBERT and Llama-3-8B-Instruct models fine-tuned on varying data partitions of the Spanish and English MASSIVE corpus and evaluated using the corresponding test set in the opposite language. The results in bold highlight the instances where Llama-3-8B-Instruct outperforms JointBERT trained with the entire dataset.



(a) Spanish Test Set

(b) English Test Set

Figure 3: Intent Accuracy (solid) and Slot F1 (dashed). Pink (Llama3) and yellow (JointBERT) represent same-language evaluation (Spanish up, English down). Blue shows cross-lingual evaluation (fine-tuned in English, evaluated in Spanish up; and vice versa down).

Note that several commonly employed efficient cross-lingual training techniques, such as incorporating the target language in the validation set, applying source data augmentation methods like paraphrasing, code-switching and machine translation, or using prompt-tuning strategies, often yield better results than those reported for JointBERT in Table 3. For a fair comparison with Llama3-8B-Instruct, none of these techniques were applied.

Fine-tuning JointBERT on the full English dataset results in an intent accuracy of only 0.1496 on the Spanish test set, whereas fine-tuning on the complete Spanish dataset and evaluating on English achieves 0.2384. In sharp contrast, Llama3-8B-Instruct exhibits remarkable cross-lingual performance, reaching an intent accuracy of 0.7576 when fine-tuned on English and evaluated on Spanish, and 0.8282 when fine-tuned on Spanish and evaluated on English, using only 5% of the training data, as highlighted in bold in Table 3. These scores are very close those achieved by the Llama3-8B-Instruct models fine-tuned and tested in the same language (see Table 1).

Figure 3 compares the performance of cross-lingual models with that of JointBERT and Llama3-8B-Instruct models in a monolingual setting. As shown, cross-lingual medium-sized LLMs demonstrate strong results, significantly outperforming JointBERT in Intent Accuracy and closely matching the performance of monolingual Llama3-8B-Instruct models. In the Slot Filling task, cross-lingual models outperform JointBERT in Spanish and in English when less than 50% of the data is used for fine-tuning. However, monolingual models consistently achieve better Slot Filling results across all cases, indicating that slot predictions are more dependent on the language.

5.3 Zero-shot Setting

Finally, two zero-shot scenarios have been evaluated in the monolingual setting using the prompt provided in Appendix B. The results are shown in Table 4 and the outcomes clearly indicate how

	Llama3-8B-Instruct		GPT-4o	
	ES	EN	ES	EN
Intent Accuracy	0.5343	0.5901	0.7905	0.7993
Slot F1	0.1625	0.1827	0.4992	0.5668
Exact Match	0.0881	0.1150	0.3712	0.4459

Table 4: Zero-shot performance on the Spanish and English MASSIVE datasets.

much better ultra-large scale models perform compared to medium-scale models for monolingual zero-shot tasks. Besides, fine-tuning the medium-scale model with just 5-10% of the data in a monolingual scenario already improves performance beyond that of the ultra-large scale model, as shown by comparing the results in Table 1 to those in 4.

6 Conclusions and Future Work

This study has explored the potential of fine-tuning a medium-scale LLM for joint IC and SF, demonstrating that models requiring only moderate hardware resources, feasible for on-premise deployment by SMEs, can deliver competitive performance with significantly reduced data compared to current leading methods. Experiments on the MASSIVE corpus have revealed that the Llama-3-8B-Instruct model, fine-tuned with just 10% of the data, outperforms both the state-of-the-art JointBERT architecture and zero-shot GPT-4o in monolingual scenarios. The study has also highlighted the strong cross-lingual performance of the Llama-3-8B-Instruct model, demonstrating that even when fine-tuned on a single language, it achieves high accuracy in other languages. This capability reduces the data required for developing NLU systems in multilingual settings, allowing companies to lower annotation costs while preserving performance.

These findings provide practical guidance for building NLU systems in data-scarce environments with limited hardware, offering best practices for creating data-efficient and cost-effective models. They are especially valuable for SMEs looking to optimize performance while managing resource constraints.

Future research directions include investigating alternative instruction formats to enhance SF performance and testing the methodology on additional medium-scale LLMs across a wider range of datasets with varying complexity levels.

References

Maia Aguirre, Ariane Méndez, Manuel Torralbo, and Arantza del Pozo. 2023. Simplifying the development of conversational speech interfaces by non-expert end-users through dialogue templates. In *International Conference on Computer-Human Interaction Research and Applications*, pages 89–109. Springer.

Bushra Altarif and Muneer Al Mubarak. 2022. Artificial intelligence: chatbot—the new generation of communication. In *Future of Organizations and Work After the 4th Industrial Revolution: The Role of Artificial Intelligence, Big Data, Automation, and Robotics*, pages 215–229. Springer.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Lynn Chua, Badih Ghazi, Yangsibo Huang, Pritish Kamath, Ravi Kumar, Pasin Manurangsi, Amer Sinha, Chulin Xie, and Chiyuan Zhang. 2024. Crosslingual capabilities and knowledge barriers in multilingual large language models. *arXiv preprint arXiv:2406.16135*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, et al. 2023. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4277–4302.

Carolina Fortuna, Din Mušić, Gregor Cerar, Andrej Čampa, Panagiotis Kapsalis, and Mihael Mohorčič. 2023. *On-Premise Artificial Intelligence as a Service for Small and Medium Size Setups*, pages 53–73. Springer International Publishing, Cham.

Mutian He and Philip N Garner. 2023. Can chatgpt detect intent? evaluating large language models for spoken language understanding. In *Interspeech*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Linmei Hu, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, and Juanzi Li. 2023. A survey of knowledge enhanced pre-trained language models. *IEEE Transactions on Knowledge and Data Engineering*.

Peng Hu, Sizhe Liu, Changjiang Gao, Xin Huang, Xue Han, Junlan Feng, Chao Deng, and Shujian Huang. 2024. Large language models are cross-lingual knowledge-free reasoners. *arXiv preprint arXiv:2406.16655*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Sang Yun Kwon, Gagan Bhatia, Elmoatez Billah Nagoudi, Alcides Alcoba Inciarte, and Muhammad Abdul-mageed. 2023. SIDLR: Slot and intent detection models for low-resource language varieties. In *Tenth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2023)*, pages 241–250, Dubrovnik, Croatia. Association for Computational Linguistics.

Sachin Mehta, Mohammad Hossein Sekhavat, Qingqing Cao, Maxwell Horton, Yanzi Jin, Chenfan Sun, Iman Mirzadeh, Mahyar Najibi, Dmitry Belenko, Peter Zatloukal, et al. 2024. Openelm: An efficient language model family with open-source training and inference framework. *arXiv preprint arXiv:2404.14619*.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Paramita Mirza, Viju Sudhi, Soumya Ranjan Sahoo, and Sinchana Ramakanth Bhat. 2024. Illuminer: Instruction-tuned large language models as few-shot intent classifier and slot filler. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8639–8651.

Soham Parikh, Mitul Tiwari, Prashil Tumbade, and Quaizar Vohra. 2023. Exploring zero and few-shot techniques for intent classification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 744–751.

GP Shrivatsa Bhargav, Sumit Neelam, Udit Sharma, Shajith Ikbal, Dheeraj Sreedhar, Hima Karanam, Sachindra Joshi, Pankaj Dhoolia, Dinesh Garg, Kyle Croutwater, et al. 2024. An approach to build zero-shot slot-filling system for industry-grade conversational assistants. *arXiv e-prints*, pages arXiv–2406.

Lifu Tu, Jin Qu, Semih Yavuz, Shafiq Joty, Wenhao Liu, Caiming Xiong, and Yingbo Zhou. 2024. Efficiently aligned cross-lingual transfer learning for conversational tasks using prompt-tuning. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1278–1294, St. Julian's, Malta. Association for Computational Linguistics.

Pei Wang, Keqing He, Yejie Wang, Xiaoshuai Song, Yutao Mou, Jingang Wang, Yunsen Xian, Xunliang Cai, and Weiran Xu. 2024. Beyond the known: Investigating llms performance on out-of-domain intent detection. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2354–2364.

Henry Weld, Xiaoqi Huang, Siqu Long, Josiah Poon, and Soyeon Caren Han. 2022. A survey of joint intent detection and slot filling models in natural language understanding. *ACM Computing Surveys*, 55(8):1–38.

Shangjian Yin, Peijie Huang, Yuhong Xu, Haojing Huang, and Jiatian Chen. 2024. Do large language model understand multi-intent spoken language? *arXiv preprint arXiv:2403.04481*.

Anis Syafiqah Mat Zailan, Noor Hasimah Ibrahim Teo, Nur Atiqah Sia Abdullah, and Mike Joy. 2023. State of the art in intent detection and slot filling for question answering system: A systematic literature review. *International Journal of Advanced Computer Science & Applications*, 14(11).

Zhihong Zhu, Xuxin Cheng, Hao An, Zhichang Wang, Dongsheng Chen, and Zhiqi Huang. 2024. Zero-shot spoken language understanding via large language models: A preliminary study. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 17877–17883.

A Data Sampling

Intents	5%	10%	20%	30%	50%	75%	100%
calendar_set	40	81	162	242	404	606	808
play_music	32	64	127	190	318	476	635
calendar_query	28	56	113	170	282	424	565
general_quirky	27	55	109	164	273	410	546
qa_factoid	27	54	108	162	270	406	541
weather_query	26	53	106	158	264	396	528
news_query	25	50	100	150	250	375	500
email_query	21	41	82	123	206	308	411
email_sendemail	18	35	71	106	176	265	353
datetime_query	15	30	61	92	152	229	305
calendar_remove	15	30	60	90	150	224	299
social_post	14	28	57	85	142	212	283
play_radio	14	28	55	83	138	207	276
qa_definition	13	27	53	80	133	200	266
transport_query	11	23	45	68	113	170	226
cooking_recipe	10	21	41	62	104	155	207
lists_query	10	19	38	57	96	143	191
play_podcasts	10	19	38	57	95	142	190
recommendation_events	9	19	37	56	93	140	186
alarm_set	9	18	36	54	90	134	179
lists_createoradd	9	17	35	52	87	130	174
recommendation_locations	8	17	34	51	85	128	170
lists_remove	8	16	32	49	81	122	162
qa_stock	8	15	30	46	76	114	152
play_audiobook	7	15	30	45	74	112	149
music_query	7	15	30	44	74	111	148
qa_currency	7	14	28	43	71	106	142
takeaway_order	7	13	26	40	66	99	132
alarm_query	6	13	26	39	65	98	130
email_querycontact	6	13	25	38	64	95	127
transport_ticket	6	13	25	38	63	94	126
iot_hue_lightoff	6	12	25	38	62	94	125
takeaway_query	6	12	24	36	60	91	121
iot_hue_lightchange	6	12	24	36	60	89	119
iot_coffee	6	12	24	36	60	89	119
transport_traffic	6	11	23	34	56	85	113
music_likeness	6	11	23	34	56	85	113
play_game	6	11	22	33	56	83	111
social_query	5	11	21	32	54	80	107
audio_volume_mute	5	10	21	31	52	77	103
audio_volume_up	5	10	20	30	50	76	101
transport_taxi	5	10	19	29	48	72	96
iot_cleaning	4	8	17	25	42	63	84
qa_maths	4	8	16	23	39	58	78
alarm_remove	4	8	15	22	38	56	75
iot_hue_lightdim	4	7	15	22	36	55	73

iot_hue_lightup	4	7	14	22	36	54	72
general_joke	3	7	14	21	34	52	69
recommendation_movies	3	7	14	20	34	51	68
email_addcontact	3	5	11	16	26	40	53
datetime_convert	3	5	10	16	26	39	52
music_settings	2	5	10	15	25	38	50
audio_volume_down	2	5	9	14	23	34	46
iot_wemo_on	2	4	8	12	20	31	41
iot_wemo_off	2	4	8	11	19	28	38
general_greet	1	2	4	7	11	16	22
iot_hue_lighton	1	2	4	6	10	16	21
audio_volume_other	1	2	4	5	9	14	18
music_dislikeness	1	1	3	4	6	10	13
cooking_query	1	1	1	1	2	3	4

Table 5: Number of utterances per intent across the different partitions

B Model Setup

B.1 JointBERT

Training parameters: num_train_epochs=850, warmup_steps=500, batch_size=128, gradient_accumulation_steps=8, learning_rate=4.7e-6, optimizer="adamw", adam_epsilon=1e-9, weight_decay=0.11.

B.2 Llama-3

Training parameters:
LoRa Config: r=16, lora_alpha=16 , lora_dropout=0.05, target_modules=["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up_proj", "down_proj", "lm_head"]

Training Arguments: num_train_epochs=50, warmup_steps=10, batch_size=512, mini_batch_size=4, gradient_accumulation_steps=batch_size/mini_batch_size, learning_rate=3e-4, optimizer="adamw_8bit", weight_decay=0.01.

For generation, the default parameters have been utilized, except for the temperature, which has been set to 0.1.

Training prompt:

```
You are an intent and entity classifier. Classify the intent and entities of this input.
 <input_sentence>
 intent: <intent_name>,
 entities: <["entity_name1: entity_value1", "entity_name2: entity_value2"]>
```

Generation prompt for the fine-tuned models:

```
You are an intent and entity classifier. Classify the intent and entities of this input.
 <input_sentence>
```

Generation prompt for the zero-shot case:

```
You are an intent and entity classifier.

Each sentence has one intent of the following list: [
    calendar_set, play_music, calendar_query, general_quirky, qa_factoid,
    weather_query, news_query, email_query, email_sendemail, datetime_query,
    calendar_remove, social_post, play_radio, qa_definition, transport_query,
    cooking_recipe, lists_query, play_podcasts, recommendation_events,
    alarm_set, lists_createoradd, recommendation_locations, lists_remove,
    qa_stock, play_audiobook, music_query, qa_currency, takeaway_order,
    alarm_query, email_querycontact, transport_ticket, iot_hue_lightoff,
    takeaway_query, iot_hue_lightchange, iot_coffee, transport_traffic,
    music_likeness, play_game, social_query, audio_volume_mute,
    audio_volume_up, transport_taxi, iot_cleaning, qa_maths, alarm_remove,
    iot_hue_lightdim, iot_hue_lightup, general_joke, recommendation_movies,
    email_addcontact, datetime_convert, music_settings, audio_volume_down,
    iot_wemo_on, iot_wemo_off, general_greet, iot_hue_lighton,
    audio_volume_other, music_dislikeness, cooking_query
]

Each sentence can have 0 or more entities. Each entity is in the format:
"entity_name: entity_value"

The entity_names must be in the following list: [
    date, time, event_name, place_name, person, media_type, business_name,
    sport_type, transport_type, weather_descriptor, food_type, relation,
    list_name, timeofday, definition_word, artist_name, device_type,
```

```
    business_type, house_place, news_topic, music_genre, player_setting,
    radio_name, currency_name, song_name, order_type, color_type, game_name,
    general_frequency, personal_info, audiobook_name, podcast_descriptor,
    meal_type, playlist_name, app_name, podcast_name, change_amount, time_zone,
    music_descriptor, joke_type, email_folder, transport_agency, email_address,
    ingredient, coffee_type, cooking_type, movie_name, movie_type,
    transport_name, alarm_type, drink_type, transport_descriptor,
    audiobook_author, game_type, music_album
]

The entity values are substrings of the input sentence.

Desired format:
intent: <intent_name>, entities: <["entity_name1: entity_value1",
"entity_name2: entity_value2"]>

Classify the intent and entities of the provided input.
```