Call Larisa Ivanovna: Code-Switching Fools Multilingual NLU Models

Alexey Birshert[(✉)] and Ekaterina Artemova

National Research University Higher School of Economics, Moscow, Russia
{abirshert,elartemova}@hse.ru
https://cs.hse.ru/en/ai/computational-pragmatics/

Abstract. Practical needs of developing task-oriented dialogue assistants require the ability to understand many languages. Novel benchmarks for multilingual natural language understanding (NLU) include monolingual sentences in several languages, annotated with intents and slots. In such setup models for cross-lingual transfer show remarkable performance in joint intent recognition and slot filling. However, existing benchmarks lack of code-switched utterances, which are difficult to gather and label due to complexity in the grammatical structure. The evaluation of NLU models seems biased and limited, since code-switching is being left out of scope.

Our work adopts recognized methods to generate plausible and naturally-sounding code-switched utterances and uses them to create a synthetic code-switched test set. Based on experiments, we report that the state-of-the-art NLU models are unable to handle code-switching. At worst, the performance, evaluated by semantic accuracy, drops as low as 15% from 80% across languages. Further we show, that pre-training on synthetic code-mixed data helps to maintain performance on the proposed test set at a comparable level with monolingual data. Finally, we analyze different language pairs and show that the closer the languages are, the better the NLU model handles their alternation. This is in line with the common understanding of how multilingual models conduct transferring between languages.

Keywords: Intent recognition · Slot filling · Code-mixing · Code-switching · Multilingual models

1 Introduction

The usability of task-oriented dialog (ToD) assistants depends crucially on their ability to process users' utterances in many languages. At the core of a task-oriented dialog assistant is a natural language understanding (NLU) component, which parses an input utterances into a semantic frame by means of intent recognition and slot filling [40]. Intent recognition tools identify user needs (such as `buy a flight ticket`). Slot filling extracts the intent's arguments (such as `departure city` and `time`).

Common approaches to training multilingual task-oriented dialogue systems rely on (i) the abilities of pre-trained language models to transfer learning across languages and [5,25] and (ii) translate-and-align pipelines [18].

The latter group of methods leverages upon pre-training on large amounts of raw textual data for many languages. Different learning objectives are used to train representations aligned across languages. The alignment can be further improved by means of task-specific label projection [27] and representation alignment [14] methods.

The former group of methods utilizes off-the-shelf machine translation engines to translate (i) the training data from resource-rich languages (almost exclusively English) to target languages or (ii) the evaluation data from target languages into English [18]. Further word alignment techniques help to match slot-level annotations [9, 30]. Finally, a monolingual model is trained to make desired predictions.

A number of datasets for cross-lingual NLU have been developed. To name a few, MultiAtis++ [44], covering seven languages across four language families, contains dialogues related to a single domain – air travels. MTOP [23] comprises six languages and 11 domains. xSID [13] is an evaluation-only small-scale dataset, collected for 12 languages and six domains.

Recent research has adopted a new experimental direction aimed at developing cross-lingual augmentation techniques, which learn the inter-lingual semantics across languages [9, 15, 21, 26, 31]. These work seek to simulate code-switching, a phenomenon where speakers use multiple languages mixed up [35]. Experimental results consistently show that augmentation with synthetic code-switched data leads to significantly improved performance for cross-lingual NLU tasks. What is more, leveraging upon these datasets in practice meets the needs of multicultural and multilingual communities. To the best of our knowledge, large-scale code-switching ToD corpora do not exist.

This paper extends the ongoing research on exploring benefits from synthetic code-switching to cross-lingual NLU. Our approach to generating code-switched utterances relies on grey-box adversarial attacks on the NLU model. We perturb the source utterances by replacing words or phrases with their translations to another language. Next, perturbed utterances are fed to the NLU model. Increases in the loss function indicate difficulties in making predictions for the utterance. This way, we can (i) generate code-switched adversarial utterances, (ii) discover insights on how code-switching with different languages impacts the performance of the target language, (iii) gather augmentation data for further fine-tuning of the language model. To sum, our contributions are[1]:

1. We implement several simple heuristics to generate code-switched utterances based on monolingual data from an NLU benchmark;
2. We show-case that monolingual models fail to process code-switched utterances. At the same time, cross-lingual models cope much better with such texts;
3. We show that fine-tuning of the language model on code-switched utterances improves the overall semantic parsing performance by up to a 2-fold increase.

---

[1] Our code can be found at https://github.com/PragmaticsLab/CodeSwitchingAdversarial.

2 Related Work

Generation of Code-Switched Text has been explored as a standalone task [17,22,33, 34,39,42] and as a way to augment training data for cross-lingual applications, including task-oriented dialog systems [9,15,21,26,31], machine translation [1,12,16], natural language inference and question answering [37], speech recognition [46].

Methods for generating code-switched text range from simplistic re-writing of some words in the target script [12] to adversarial attacks on cross-lingual pre-trained language models [39] and building complex hierarchical VAE-based (Variational AutoEncoders) models [34]. The vast majority of methods utilize machine translation engines [37], parallel datasets [1,12,16,42] or bilingual lexicons [39] to replace the segment of the input text with its translations. Bilingual lexicons may be induced from the parallel corpus with the help of soft alignment, produced by attention mechanisms [22,26]. Pointer networks can be used to select segments for further replacement [17,42]. If natural code-switched data is available, such segments can be identified with a sequence labeling model [16]. Other methods rely on linguistic theories of code-switching. To this end, GCM toolkit [33] leverages two linguistic theories, which help to identify segments where code-switching may occur by aligning words and mapping parse trees of parallel sentences.

The quality of generated code-switched texts is evaluated by (i) **intrinsic** text properties, such as code-switching ratio, length distribution, and (ii) **extrinsic** measures, ranging from the perplexity of external languages model to the downstream task performance, in which code-switched data was used for augmentation [34].

Natural Language Understanding in the ToD domain has two main goals, namely intent recognition, and slot filling [32]. Intent recognition assigns a user utterance with one of the pre-defined intent labels. Thus, intent recognition is usually tackled with classification methods. Slot filling seeks to find arguments of the assigned intent and is modeled as a sequence modeling problem. For example, the utterance *I need a flight from Moscow to Tel Aviv on 2nd of December* should be assigned with the intent label `find flight`; three slots may be filled: `departure city`, `arrival city`, `date`. These two interdependent NLU tasks are frequently approached via multitask learning, in which a joint model is trained to recognize intents and fill in slots simultaneously [3,20,43].

Adversarial Attacks on natural language models has been categorized with respect to (i) what kind of information is provided from the model and (ii) what kind of perturbation is applied to the input text [29]. White-box attacks [8] have access to the whole model's inner workings. On the opposite side, black-box attacks [11] do not have any knowledge about the model. Grey-box attacks [45] access predicted probabilities and loss function values. Perturbations can be applied at char-, token-, and sentence-levels [4,11,24].

Other Related Research Directions include **code-switching detection** [28,38], evaluation of pre-trained language models' **robustness** to code-switching [41], analysis of

language model's inner workings with respect to code-switched inputs [36], **benchmarking** downstream tasks in code-switched data [2, 19].

3   Our Approach

In our work we train multilingual language models for the joint intent recognition and slot-filling task.

3.1   Dataset

We chose MultiAtis++ dataset [44] as the main source of data. This dataset contains seven languages from three language families - Indo-European (English, German, Portuguese, Spanish, and French), Japanese, and Sino-Tibetan (Chinese). The dataset is a parallel corpus for classifying intents and slot labeling - in 2020 it was translated from English to the other six languages. The training set contains 4978 samples for each language; the test set contains 893 samples per language. Each object in the dataset consists of a sentence, slot labels in BIO format, and the intent label.

3.2   Joint Intent Recognition and Slot-Filling

We train a single model for the joint intent recognition and slot-filling task. The model has two heads; the first one predicts intents, and the second one predicts slots. We had trained two different models as a backbone - m-BERT and XLM-RoBERTa.

We aim at comparing two setups: (i) training on the whole dataset and (ii) only on its English subset followed by zero-shot inference for other languages. For convenience, we propose short names for the four models trained during the research - **xlm-r, xlm-r-en, m-bert, m-bert-en**.

We measure our models' quality with three metrics: **intent accuracy**, **F1 score** for slots (we used micro-averaging by classes) and the proportion of sentences where we correctly classified everything - both intent and all slots - **semantic accuracy**.

3.3   Code-switching Generation

We propose two variants of gray-box adversarial attacks. During the attack, we have access to the model's loss of input data. We strive to create an attack so that the resulting adversarial perturbation of the source sentence is as close as possible to the realistic code-switching. Quality evaluation at such adversarial attacks can act as a lower bound for corresponding models' quality in similar problems in the presence of real code-switching in input data.

We focus mainly on the lexical aspect of code-switching when some words are replaced with their substitutes from other languages. We replace some tokens in the source sentence with their equivalents from the attacking languages during the attack. The method to determine the replacement depends on which exactly attack is used. Since most people who can use code-switching are bilinguals, in our work, we propose to analyze attacks consisting in embedding one language into another.

Algorithm 1. General view of the attack

Require: Sentence and label x, y; source model $\mathcal{M}$; embedded target language L
Ensure: Adversarial sample x'
  $\mathcal{L}_x$ = GetLoss($\mathcal{M}$, x, y)
  for i in permutation(len(x)) **do**
      Candidates = GetCandidates($\mathcal{M}$, x, y, token_id = i)
      Losses = GetLoss($\mathcal{M}$, Candidates)
      if Candidates and max(Losses) $\geq \mathcal{L}_x$ **then**
          $\mathcal{L}_x$ = max(Losses)
          x, y = Candidates[argmax(Losses)]
      end if
  end for
      return x

Overview of the Attacks. The general attack scheme (Algorithm 1) is the same for both proposed attacks. We offer the following attack pattern in our work: a source model, a pair of sample sentences and labels, and embedded target language. Then we iterate over the tokens in the sample sentence and strive to replace them with their equivalent from the embedded target language. If changing the token to its equivalent increases the source model's loss, we replace the token with the proposed candidate. The difference between the two methods consists in the way they generate replacement candidates.

Word-level Adversaries. The first attack (Algorithm 2) generates target embedded language substitutions by translating single tokens into the corresponding language. Attacking this way, we make a rough lower bound since we do not consider the context of the sentences and the ambiguity of words during the attack. To translate words into other languages, we use the large-scale many-to-many machine translation model M2M-100 from Facebook [10]. You can see an example of this attack in table 1.

Algorithm 2. Word-level attack

Require: Machine translation model T
  function GETCANDIDATES($\mathcal{M}$, x, y, token_id)
      if x[token_id] in $T[L]$ **then**
          tokens = $T[L]$[x[token_id]]
          x[token_id] = tokens
          y[token_id] = ExtendSlotLabels(y[token_id], len(tokens))
      end if
      return x, y
  end function

Table 1. Example of attacking XLM-RoBERTa (xlm-r) with word-level attack.

Utterance en	What are the flights from las vegas to Ontario
Utterance adv	What sind die flights from las vegas to Ontario

Phrase-level Adversaries. The second attack (Algorithm 3) generates equivalents from other languages by building alignments between sentences in different languages. One sentence is a translation of another; we utilize the fact that we have a parallel dataset. Candidates for each token are defined as tokens from the embedded sentence into which the token was aligned. For aligning sentences, we use the awesome-align model based on m-BERT [7]. You can see an example of this attack in table 2.

Algorithm 3. Phrase-level attack

Require: Sentences alignment A
  function GETCANDIDATES($\mathcal{M}$, x, y, token_id)
    if x[token_id] in $A[L]$ **then**
      tokens = $A[L]$[x[token_id]]
      x[token_id] = tokens
      y[token_id] = ExtendSlotLabels(y[token_id], len(tokens))
    end if
    return x, y
  end function

Table 2. Example of attacking XLM-RoBERTa (xlm-r) with phrase-level attack.

Utterance en	Please find flights available from kansas city to newark
Utterance adv	Encontre find flights disponíveis from kansas city para newark

3.4 Adversarial Pre-training Method Protects from Adversarial Attacks

Adversarial pre-training protects the model against the proposed adversarial attacks. It most likely allows the model to increase the performance not only at adversarial perturbations but also on real data with code-switching. However, this is only a hypothesis since there is no real-life code-switched ToD data.

The adversarial pre-training method relies on domain adaptation techniques and has several steps:

1. Generating adversarial training set for masked language modeling task.
2. Fine-tuning language model's body on the new generated set in masked language modeling task.
3. Loading fine-tuned model's body before training for joint intent classification and slot labeling task.

Generating Adversarial Training Set. To generate an adversarial training set, we use an adaptation of the phrase-level algorithm of the adversarial attack (Algorithm 4). The difference is that tokens are replaced with their equivalents with the probability of 0.5. Thus, a trained model is not required to generate the sample. The adversarial training set is a concatenation of generated sets for all languages in the dataset except English. Each subset is generated by embedding the target language into the training set of the MultiAtis++ dataset in English. After generation, we get six subsets of 4884 sentences each. The final adversarial training set consists of 29304 sentences; we divide it into training and test sets in a ratio of 9 to 1.

Algorithm 4. Generating adversarial training set

Require: Training dataset X, set of embedded languages $L_1, \ldots L_n$
Ensure: Adversarial training set X'

  X' = [ ]
  for L in $L_1, \ldots L_n$ **do**
    for x in X **do**
      for i in permutation(len(x)) **do**
        Candidates = GetCandidates($\mathcal{M}$, x, y, token_id = i)
        if Candidates and $\mathcal{U}(0, 1)$ ¿ 0.5 **then**
          x, _ = random.choice(Candidates)
        end if
      end for
      X'.append(x)
    end for
  end for
  return X'

Fine-tuning Model's Body. After generating the adversarial training set, we fine-tune the pre-trained multilingual model. The model is trained with the masked language modeling objective [6]. We select 15% of tokens and predict them using the model to train a model for such a task. 80% of the selected tokens are replaced with the mask token, 10% are replaced with random words from the model's dictionary, the remaining 10% remain unchanged. After fine-tuning, we dump the body of the model for future use.

Loading Fine-Tuned Model's Body. Before training the multilingual model for the task of join intent classification and slot labeling, we load the fine-tuned body of the model. For models that have been pre-trained using the adversarial pre-training method, we will add the suffix **adv** to the name.

4 Experimental Results

We will compare models trained only on the English training set (zero-shot models) and the whole training set (full models). We will evaluate the quality by three metrics -

accuracy for intents, f1-score for slots and semantic accuracy. We found that zero-shot models have significantly worse quality than full ones, not only in languages other than English but even in English.

Joint Intent Classification and Slot Labelling. We have achieved strong performance for the problem of classifying intents and filling in slots. On the test sample, full models showed an average of 97% correct answers for intents, and zero-shot ones, on average, 85%. Full models showed 0.93 f1-score for slots, zero-shot 0.68. Full models showed 79% of completely correctly classified sentences and zero-shot ones - about 26%. This shows that zero-shot learning is not capable of competing with full learning in this particular task. In the Fig 1, you can see the comparison between models and languages by Intent accuracy metric. The additional results are provided in the project's repository.
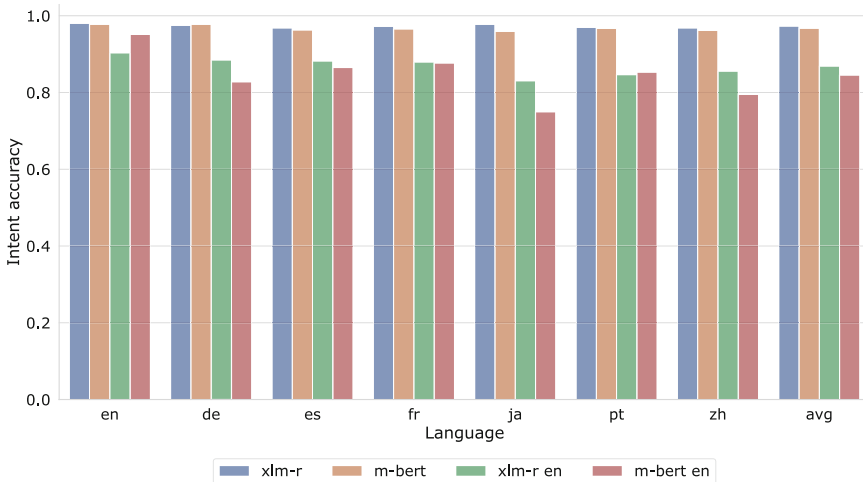

Fig. 1. Model comparison by **Intent accuracy** metric.

Attacking Models. We attacked all the models using our two algorithms. We found that the word-level attack turned out to be more advanced, leading to lower performance. For full models, the quality of intents fell from 98% to 88%, for zero-shot models from 92% to 77%. For full models, the quality by slots fell from 0.95 to 0.6, for zero-shot models from 0.88 to 0.48. For full models, the proportion of entirely correctly classified sentences fell from 83% to 14%, for zero-shot ones from 60% to 5%. Figure 2 compares results before and after the word-level attack with the Intent accuracy metric.

We also got that the phrase-level attack turned out to be softer and gave a higher quality compared to the word-level attack. For full models, the quality of intents fell from 98% to 95%, for zero-shot models from 92% to 80%. For full models, the quality by slots fell from 0.95 to 0.7, for zero-shot models from 0.88 to 0.55. For full models,

the proportion of entirely correctly classified sentences fell from 83% to 35%, for zero-shot ones from 60% to 10%. In the Fig 3, you can see the quality comparison after the phrase-level attack for the Intent accuracy metric.
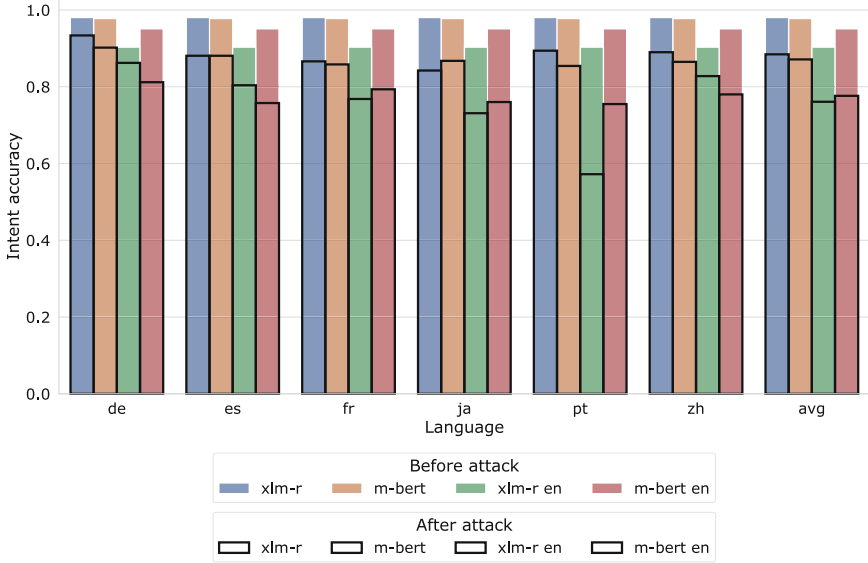

Fig. 2. Model comparison after **word-level** attack by **Intent accuracy** metric.

Adversarial Pre-training. To protect the models from attacks, we fine-tuned the bodies for both models and loaded them before training for the task of joint intent classification and slot filling. We found that the defense had almost no effect on the full models in terms of quality on the test set. For zero-shot models, the effect on the test set is ambiguous - the quality by intents fell for Asian languages but increased slightly for all others. As for the slots, we observe a negative effect for the m-BERT model and a positive effect for the XLM-RoBERTa model.

For the word-level attack, a slight deterioration in the quality of intents for Asian languages is noticeable, and a positive effect for other languages. After the adversarial pre-training, the quality of slots increased for all models, which ultimately results in an almost two-fold increase in the proportion of entirely correctly classified sentences for zero-shot models and about 15% relative improvement for full models.
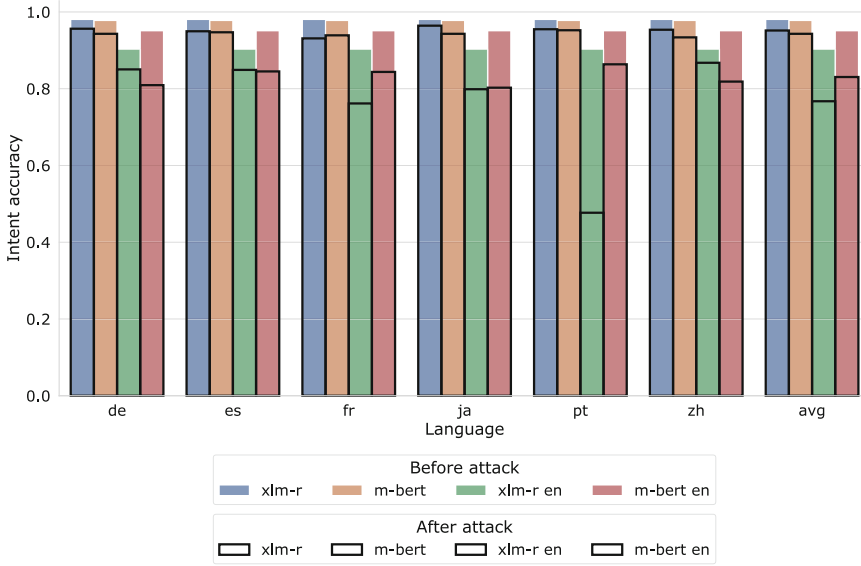
Fig. 3. Model comparison after **phrase-level** attack by **Intent accuracy** metric.
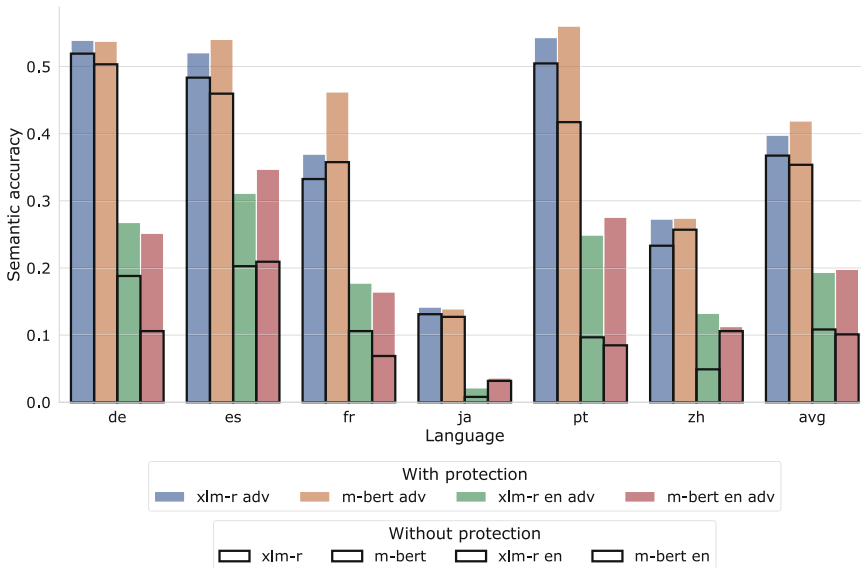

Fig. 4. Model comparison **with protection** after **phrase-level** attack by **Semantic accuracy** metric.

Again, for the phrase-level attack, a slight deterioration in the quality of intents for Asian languages is noticeable, and a positive effect for other languages. After the defense, the quality of slots dropped slightly for Asian languages and increased significantly for the rest. This results in a two-fold increase in the proportion of entirely correctly classified sentences for zero-shot models and about 15% relative improvement for full models (Fig 4).

5   Discussion

We approached the problem of recognizing intents and filling in slots for a multi-lingual ToD system. We study the effect of switching codes on two multi-lingual language models, XLM-RoBERTa and m-BERT. We showed that switching codes could become a noticeable problem when applying language models in practice using two gray-box attacks. However, the defense method shows promising results and helps to improve the quality after the model is attacked.

6   Conclusion

This paper presents an adversarial attack on multilingual task-oriented dialog (ToD) systems that simulates code-switching. This work is motivated by research and practical needs. First, the proposed attack reveals that pre-trained language models are vulnerable to synthetic code-switching. To this end, we develop a simplistic defense technique against code-switched adversaries. Second, our work is motivated by the practical needs of multilingual ToDs to cope with code-switching, which is seen as an essential phenomenon in multicultural societies. Future work directions include evaluating how plausible and naturally-sounding code-switched adversaries are and adopting similar approaches to model-independent black-box scenarios.

References

1. Abdul-Mageed, M., Lakshmanan, L.V.: Exploring text-to-text transformers for English to Hinglish machine translation with synthetic code-mixing. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 36–46 (2021)
2. Aguilar, G., Kar, S., Solorio, T.: LinCE: A centralized benchmark for linguistic code-switching evaluation. In: Proceedings of The 12th Language Resources and Evaluation Conference. pp. 1803–1813. European Language Resources Association, Marseille, France (2020). https://www.aclweb.org/anthology/2020.lrec-1.223
3. Chen, Q., Zhuo, Z., Wang, W.: BERT for joint intent classification and slot filling (2019)
4. Cheng, M., Yi, J., Chen, P.Y., Zhang, H., Hsieh, C.J.: Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3601–3608 (2020)

5. Conneau, A.: Unsupervised cross-lingual representation learning at scale(2019). arXiv preprint arXiv:1911.02116

6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: 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, vol. 1 (Long and Short Papers), pp. 4171–4186 (2019)

7. Dou, Z.Y., Neubig, G.: Word alignment by fine-tuning embeddings on parallel corpora. In: Conference of the European Chapter of the Association for Computational Linguistics (EACL) (2021)

8. Ebrahimi, J., Rao, A., Lowd, D., Dou, D.: Hotflip: White-box adversarial examples for text classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, vol. 2, pp. 31–36 (2018)

9. Einolghozati, A., Arora, A., Lecanda, L.S.M., Kumar, A., Gupta, S.: El volumen louder por favor: code-switching in task-oriented semantic parsing. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pp. 1009–1021 (2021)

10. Fan, A., et al.: Beyond English-centric multilingual machine translation (2020). arXiv preprint

11. Gao, J., Lanchantin, J., Soffa, M.L., Qi, Y.: Black-box generation of adversarial text sequences to evade deep learning classifiers. In: 2018 IEEE Security and Privacy Workshops (SPW), pp. 50–56, IEEE (2018)

12. Gautam, D., Kodali, P., Gupta, K., Goel, A., Shrivastava, M., Kumaraguru, P.: Comet: Towards code-mixed translation using parallel monolingual sentences. In: Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching, pp. 47–55 (2021)

13. van der Goot, R., et al.: From masked language modeling to translation: Non-english auxiliary tasks improve zero-shot spoken language understanding (2021)

14. Gritta, M., Iacobacci, I.: Xeroalign: Zero-shot cross-lingual transformer alignment (2021). arXiv preprint arXiv:2105.02472

15. Guo, Y.: Learning from multiple noisy augmented data sets for better cross-lingual spoken language understanding (2021). arXiv preprint arXiv:2109.01583

16. Gupta, A., Vavre, A., Sarawagi, S.: Training data augmentation for code-mixed translation. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5760–5766 (2021)

17. Gupta, D., Ekbal, A., Bhattacharyya, P.: A semi-supervised approach to generate the code-mixed text using pre-trained encoder and transfer learning. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, pp. 2267–2280 (2020)

18. Hu, J., Ruder, S., Siddhant, A., Neubig, G., Firat, O., Johnson, M.: Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In: International Conference on Machine Learning, PMLR, pp. 4411–4421 (2020)

19. Khanuja, S., Dandapat, S., Srinivasan, A., Sitaram, S., Choudhury, M.: Gluecos: An evaluation benchmark for code-switched NLP. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 3575–3585 (2020)

20. Kim, Y., Kim, D., Kumar, A., Sarikaya, R.: Efficient large-scale neural domain classification with personalized attention. In: Gurevych, I., Miyao, Y. (eds.) Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15–20, vol. 1, pp. 2214–2224, Association for Computational Linguistics (2018). 10.18653/v1/P18-1206, https://aclanthology.org/P18-1206/

21. Krishnan, J., Anastasopoulos, A., Purohit, H., Rangwala, H.: Multilingual code-switching for zero-shot cross-lingual intent prediction and slot filling (2021). arXiv preprint arXiv:2103.07792

22. Lee, G., Li, H.: Modeling code-switch languages using bilingual parallel corpus. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 860–870 (2020)

23. Li, H., Arora, A., Chen, S., Gupta, A., Gupta, S., Mehdad, Y.: MTOP: a comprehensive multilingual task-oriented semantic parsing benchmark. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume. pp. 2950–2962 (2021)

24. Li, L., Ma, R., Guo, Q., Xue, X., Qiu, X.: BERT-attack: Adversarial attack against BERT using BERT. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 6193–6202 (2020)

25. Liu, Y., et al.: Multilingual denoising pre-training for neural machine translation. Trans. Assoc. Comput. Linguis. **8**, 726–742 (2020)

26. Liu, Z., Winata, G.I., Lin, Z., Xu, P., Fung, P.N.: Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. In: Proceedings of the AAAI Conference on Artificial Intelligence (2020)

27. Liu, Z., Winata, G.I., Xu, P., Lin, Z., Fung, P.: Cross-lingual spoken language understanding with regularized representation alignment. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 7241–7251 (2020)

28. Mave, D., Maharjan, S., Solorio, T.: Language identification and analysis of code-switched social media text. In: Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching. pp. 51–61, Association for Computational Linguistics, Melbourne, Australia (2018). 10.18653/v1/W18-3206, https://aclanthology.org/W18-3206

29. Morris, J., Lifland, E., Yoo, J.Y., Grigsby, J., Jin, D., Qi, Y.: Textattack: a framework for adversarial attacks, data augmentation, and adversarial training in NLP. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 119–126 (2020)

30. Nicosia, M., Qu, Z., Altun, Y.: Translate & fill: Improving zero-shot multilingual semantic parsing with synthetic data (2021). arXiv preprint arXiv:2109.04319

31. Qin, L., Ni, M., Zhang, Y., Che, W.: CoSDA-ML: Multi-lingual code-switching data augmentation for zero-shot cross-lingual NLP (2020). arXiv preprint arXiv:2006.06402

32. Razumovskaia, E., Glavaš, G., Majewska, O., Ponti, E.M., Korhonen, A., Vulić, I.: Crossing the conversational chasm: A primer on natural language processing for multilingual task-oriented dialogue systems (2021)

33. Rizvi, M.S.Z., Srinivasan, A., Ganu, T., Choudhury, M., Sitaram, S.: GCM: A toolkit for generating synthetic code-mixed text. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, pp. 205–211 (2021)

34. Samanta, B., Reddy, S., Jagirdar, H., Ganguly, N., Chakrabarti, S.: A deep generative model for code-switched text (2019). arXiv preprint arXiv:1906.08972

35. Sankoff, D., Poplack, S.: A formal grammar for code-switching. Res. Lang. & Soc. Interac. **14**(1), 3–45 (1981)

36. Santy, S., Srinivasan, A., Choudhury, M.: BERTologicomix: How does code-mixing interact with multilingual BERT? In: Proceedings of the Second Workshop on Domain Adaptation for NLP, pp. 111–121 (2021)

37. Singh, J., McCann, B., Keskar, N.S., Xiong, C., Socher, R.: XLDA: Cross-lingual data augmentation for natural language inference and question answering (2019). arXiv preprint arXiv:1905.11471

38. Sravani, D., Kameswari, L., Mamidi, R.: Political discourse analysis: a case study of code mixing and code switching in political speeches. In: Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching. pp. 1–5, Association for Computational Linguistics (2021). 10.18653/v1/2021.calcs-1.1, https://aclanthology.org/2021.calcs-1.1
39. Tan, S., Joty, S.: Code-mixing on sesame street: dawn of the adversarial polyglots. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 3596–3616 (2021)
40. Tur, G., Hakkani-Tür, D., Heck, L.: What is left to be understood in ATIS? In: 2010 IEEE Spoken Language Technology Workshop, IEEE, pp, 19–24 (2010)
41. Winata, G.I., Cahyawijaya, S., Liu, Z., Lin, Z., Madotto, A., Fung, P.: Are multilingual models effective in code-switching? NAACL **2021**, 142 (2021)
42. Winata, G.I., Madotto, A., Wu, C.S., Fung, P.: Code-switched language models using neural based synthetic data from parallel sentences. In: Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), pp. 271–280 (2019)
43. Wu, C.S., Hoi, S.C., Socher, R., Xiong, C.: TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 917–929. Association for Computational Linguistics (2020). 10.18653/v1/2020.emnlp-main.66, https://aclanthology.org/2020.emnlp-main.66
44. Xu, W., Haider, B., Mansour, S.: End-to-end slot alignment and recognition for cross-lingual NLU. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 5052–5063 (2020)
45. Xu, Y., Zhong, X., Yepes, A.J., Lau, J.H.: Grey-box adversarial attack and defence for sentiment classification. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4078–4087 (2021)
46. Yılmaz, E., Heuvel, H.v.d., van Leeuwen, D.A.: Acoustic and textual data augmentation for improved asr of code-switching speech (2018). arXiv preprint arXiv:1807.10945