Octopus:
A Multitask Model and Toolkit for Arabic Natural Language Generation

AbdelRahim Elmadany[ξ,⋆] **El Moatez Billah Nagoudi**[ξ,⋆] **Muhammad Abdul-Mageed**[ξ,λ,⋆]
[ξ] Deep Learning & Natural Language Processing Group, The University of British Columbia
[λ]Department of Natural Language Processing & Department of Machine Learning, MBZUAI
`{a.elmadany,moatez.nagoudi,muhammad.mageed}@ubc.ca`

Abstract

Understanding Arabic text and generating human-like responses is a challenging endeavor. While many researchers have proposed models and solutions for individual problems, there is an acute shortage of a comprehensive Arabic natural language generation toolkit that is capable of handling a wide range of tasks. In this work, we present a novel Arabic text-to-text Transformer model, namely AraT5$_{\text{v2}}$. Our new model is methodically trained on extensive and diverse data, utilizing an extended sequence length of $2,048$ tokens. We explore various pretraining strategies including unsupervised, supervised, and joint pertaining, under both single and multitask settings. Our models outperform competitive baselines with large margins. We take our work one step further by developing and publicly releasing OCTOPUS, a Python-based package and command-line toolkit tailored for *eight* Arabic generation tasks all exploiting a *single* model. We release the models and the toolkit on our public repository.[1]

1 Introduction

Natural Language Generation (NLG) is a fundamental component of natural language processing that aims to generate human-like, coherent, contextually fitting, and linguistically precise text from structured data or various other input formats. NLG systems find applications in various aspects of daily life, including education, healthcare, business, and more. The recent emergence of generative models has significantly impacted the field of NLG. While important progress has been made in NLG research, the majority of existing tools, systems, and models are primarily focused on English (Jhaveri et al., 2019; Khan et al., 2021; Lauriola et al., 2022), leaving behind many languages, including Arabic.

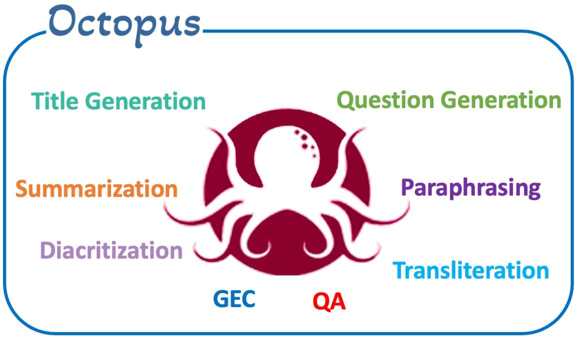

Figure 1: OCTOPUS is a jointly pretrained to cover eight NLG tasks, all shown in the illustration.

Although it is one of the most widely spoken languages in the world, and one with a rich linguistic structure and diverse dialects, Arabic remains underrepresented in NLG. One reason is the complex morphology and syntax of Arabic. Hence, the primary focus of our research here is to develop an advanced tool capable of performing several key Arabic NLG tasks. For example, we target tasks such as *text summarization*, *question answering*, *question generation*, *news headline generation*, and *paraphrasing*. These are tasks that necessitate a deep understanding of semantics, syntax, and pragmatics of Arabic. We also focus on tasks that require an understanding of both the syntax and morphology such as *diacritization*, *transliteration*, and *grammatical error correction*. Our main contributions are as follows:

1. We pretrain better and faster-to-converge versions of the text-to-text transformer model AraT5, collectively dubbed *AraT5$_{v2}$*. Compared to Nagoudi et al. (2022b), we train these new versions on a larger and more diverse dataset, as well as a larger sequence length.

2. To develop our models, we investigate diverse training strategies that integrate a combination of *supervised* and *unsupervised* training techniques.

[1] https://github.com/UBC-NLP/octopus
⋆Equal contributions

Proceedings of the The First Arabic Natural Language Processing Conference (ArabicNLP 2023), pages 232–243
December 7, 2023 ©2023 Association for Computational Linguistics

3. We introduce OCTOPUS, a Python-based toolkit for eight Arabic NLG tasks. Our tool can be used as a strong baseline or as a core enabling technology that facilitates other developments.

4. We will make OCTOPUS publicly available to the research community.

2 Related Work

In the following section, we offer a concise overview of publicly available Arabic NLU and NLG tools, along with the Arabic and multilingual sequence-to-sequence (S2S) language models that we employ in this work.

2.1 Arabic NLP Tools

NLU tools. Numerous attempts have been made to develop tools for assisting with Arabic. Some tools focus on aspects such as morphosyntax, encompassing tasks like morphological analysis, disambiguation, part-of-speech tagging, and diacritization. Notable examples include Stanford CoreNLP (Manning et al., 2014), MADAMIRA (Pasha et al., 2014), Farasa (Darwish and Mubarak, 2016), and CAMeL tools (Obeid et al., 2020). Other tools, such as Mazajek (Farha and Magdy, 2019), and AraNet (Abdul-Mageed et al., 2019), are dedicated to social meaning tasks such as sentiment analysis, emotion detection, age and gender prediction, and sarcasm detection.
NLG Tools. Regarding Arabic NLG, as far as we know, the only publicly available tools are primarily focused on many-to-Arabic machine translation (MT). These include OPEN-MT (Tiedemann and Thottingal, 2020), NLLB (Costa-jussà et al., 2022), and Turjuman (Nagoudi et al., 2022d).

2.2 Arabic S2S Language Model.

Here, we overview the Arabic sequence-to-sequence models we employ as baseline in this work.
mT5. This is the multilingual version of T5 model (Raffel et al., 2019) introduced by Xue et al. (2020). Pretraining of mT5 is performed on the extensive mC4 (Multilingual Colossal Clean Crawled Corpus) which covers 101 languages, including Arabic.
mT0. Developed by Muennighoff et al. (2022), this is a group of S2S models ranging from 300M to 13B parameters trained to investigate cross-lingual generalization through multitask fine-tuning. The models are finetuned from pre-existing mT5 (Xue et al., 2020) multilingual language models using a cross-lingual task mixture called xP3.
AraBART. Introduced by (Eddine et al., 2022), this is a pretrained encoder-decoder model designed specifically for abstractive summarization tasks in the Arabic language. AraBART follows the architecture of BART (Lewis et al., 2019a) and has been pretrained on a 73GB of Arabic text data.
AraT5. Presented by Nagoudi et al. (2022c), this is an Arabic text-to-text Transformer model dedicated to MSA and Arabic dialects. It is similar in configuration and size to T5 (Raffel et al., 2019) and is trained on 248GB of Arabic text (70GB MSA and 178GB tweets). We now introduce our new model.

3 AraT5$_{\text{v2}}$

In this section, we present a novel version of AraT5, the Arabic-specific sequence-to-sequence model. We refer to this novel version as AraT5$_{\text{v2}}$. This new version represents a substantial evolution of the original AraT5$_{\text{v1}}$ model,[2] marked by notable improvements. These include **(1)** training on an expanded dataset comprising both labeled and unlabeled data, **(2)** larger sequence length of $2,048$ tokens, and **(3)** diverse training strategies that integrate a combination of unsupervised and supervised training techniques. Table 1 provides a comparison between AraT5$_{\text{v1}}$ and AraT5$_{\text{v2}}$.
Pretraining data. As we mentioned previously, our pretraining (unlabeled and labeled) dataset is linguistically diverse, covering all categories of Arabic (i.e., CA, DA, and MSA). as we will now describe.

3.1 Unlabled Data

We collect approximately 250GB of Arabic MSA text, which corresponds to around 25.6B tokens.[3] We use different sources including AraNews$_{\text{v2}}$ (Nagoudi et al., 2020), El-Khair (El-Khair, 2016), Gigaword,[4] OSIAN (Zeroual et al., 2019), Wikipedia Arabic, Hindawi Books,[5] OSCAR$_{\text{Egyptian}}$ (Suárez et al., 2019), and AraC4 (Nagoudi et al., 2022a).[6] To obtain Classical Arabic (CA) data, we utilize the Open Islamicate Texts Initiative (OpenITI) corpus (v1.6) (Nigst et al., 2020). The OpenITI corpus consists of 11K

[2] In this paper, we refer to the original AraT5 (Nagoudi et al., 2022b) as AraT5$_{\text{v1}}$.

[3] We note that AraT5$_{\text{v1}}$ trained only on 70GB MSA data.

[4] https://catalog.ldc.upenn.edu/LDC2009T30.

[5] https://www.hindawi.org/books.

[6] We note that AraC4 contains a diverse Arabic dialect as described in (Nagoudi et al., 2022a).

	AraT5$_{\text{v1}}$	AraT5$_{\text{v1}}$-MSA	AraT5$_{\text{v1}}$-TWT	AraT5$_{\text{v2}}$
Data size	248 GB	70 GB	178 GB	250 GB
Tokens count	29 B	7.1 B	21.9 B	25.6 B
Linguistic diversity	MSA, Tweets†	MSA	Tweets†	CA, DA, MSA
Sequence length	512	512	512	$2,048$

Table 1: Comparison between AraT5$_{\text{v1}}$ and AraT5$_{\text{v2}}$ models. It is worth noting that our new model (AraT5$_{\text{v2}}$) does not include tweets, whereas 71.77% of AraT5$_{\text{v1}}$ data is from Twitter (with the remaining 28.23% sourced from other sources). **CA:** Classical Arabic. **DA:** Dialectical Arabic. **MSA:** Modern Standard Arabic. Notably, Tweets† may encompass content in CA, DA, and MSA.

Islamic books, primarily collected from sources such as Shamela Library,[7] Al-Jami Al-Kabir collection (JK),[8] books digitized by the Jordanian publisher Markaz Al-Turāth, and the Shia Library.[9]

3.2 Labeled Data

Recently, Nagoudi et al. (2023) introduced *Dolphin*, an NLG benchmark for Arabic. Dolphin covers MSA, Classical Arabic, and various Arabic dialects. It is composed of 40 datasets, making it the largest and most diverse Arabic NLG benchmark. Due to the availability of the powerful Arabic machine translation toolkit, TURJUMAN (Nagoudi et al., 2022d), we shift our focus away from machine translation, code-switching, and Arabization tasks in this paper. Hence, we utilize datasets from eight out of the total thirteen NLG tasks in Dolphin. In the following sections, we will provide a brief description of each of these tasks.

(1) Diacritization. Is the computational procedure of adding missing diacritics or vowels to Arabic texts. For this task, we use the Arabic diacritization dataset presented by Fadel et al. (2019).

(2) Grammatical Error Correction. The GEC task is centered around the analysis of written text with the aim of automatically identifying and correcting a range of grammatical errors. We use three GEC datasets: QALB 2014 (Mohit et al., 2014), QALB 2015 (Rozovskaya et al., 2015), and ZAEBUC (Habash and Palfreyman, 2022).

(3) News Title Generation. The objective of this task is to generate a suitable headline for a given news article. To accomplish this, we use two datasets: Arabic NTG (Nagoudi et al., 2022c) and XLSum (Hasan et al., 2021).[10]

(4) Paraphrasing. In this task, we use four paraphrasing datasets: AraPara, a multi-domain Arabic paraphrase dataset (Nagoudi et al., 2022c), ASEP, an Arabic SemEval paraphrasing dataset (Cer et al., 2017), Arabic paraphrasing benchmark (APB) (Alian et al., 2019), and TaPaCo (Scherrer, 2020).[11]

(5) Question Answering. In this task, four publicly available extractive QA datasets are employed: ARCD (Mozannar et al., 2019) and the Arabic part of the following three multilingual datasets: MLQA (Lewis et al., 2019b), XQuAD (Artetxe et al., 2020), and TyDiQA (Artetxe et al., 2020).

(6) Question Generation. The goal of this task is to create simple questions that are pertinent to passages, along with their corresponding answers. For this, we utilize triplets consisting of *passages*, *answers*, and *questions*, all extracted from the same QA datasets.

(7) Text Summarisation. This task includes five publicly available datasets, including both Arabic and multilingual data: MassiveSum (Varab and Schluter, 2021), XLSum Hasan et al. (2021), CrossSum (Bhattacharjee et al., 2021), ANT (Chouigui et al., 2021), and MarSum (Gaanoun et al., 2022).

(8) Transliteration. This task involves converting words or text from one writing system to another while maintaining the original language's pronunciation and sound. Three datasets are used to create this component: ANETA (Ameur et al., 2019), ATAR (Talafha et al., 2021), and NETransliteration (Merhav and Ash, 2018).

4 Training Strategies

In this section, we describe the different strategies we use to pretrain and finetune AraT5$_{\text{v2}}$.

4.1 Unsupervised Pretraining.

Here, we focus on using only our unlabeled data (see Section 3.1) for pretraining our AraT5$_{\text{v2}}$.

[7] https://shamela.ws.

[8] http://kitab-project.org/docs/openITI.

[9] https://shiaonlinelibrary.com.

[10] We note that XLSum (Hasan et al., 2021) contains news articles that are annotated with both summaries and titles. For the NTG task, we use the pairs of articles and titles used to create the training data.

[11] We use the Arabic part only of TaPaCo.

The objective function does not rely on labels but instead imparts the model with transferable knowledge that can be effectively applied to various downstream tasks. We follow Raffel et al. (2019) in using a masked language modeling "*span-corruption*" objective. This approach involves replacing consecutive spans of input tokens with a mask token, and the model is trained to reconstruct the masked tokens.

4.2 Supervised Finetuning

We use the labeled data (see Section 3.2) to finetune the AraT5$_{v2}$ models under two settings: (i) *single task* and (ii) *multitask* finetuning.

Single task finetuning. We individually finetune our AraT5$_{v2}$ models on each of the eight NLG tasks we select from the Dolphin NLG benchmark (Nagoudi et al., 2023).

Multitask finetuning. We additionally explore multitask learning (Caruana, 1997; Ruder, 2017) using our AraT5$_{v2}$ models. This strategy involves training the model on several tasks concurrently, allowing the model and its parameters to be shared across all tasks. The ultimate goal is to enhance performance on each individual task over time. To indicate the intended task for the model, we incorporate a task-specific text "*prefix*" to the original input sequence before it is fed into the model. For example, for the paraphrase task, the source will be: *paraphrase:* امرأة تضيف التوابل إلى اللحم. The model should predict إمرأة تضيف المكونات إلى لحم البقر.

4.3 Joint Pretraining and Finetuning

In this scenario, we establish a uniform training objective for both pretraining and finetuning. The model is trained using a maximum likelihood objective, employing "*teacher forcing*" (Raffel et al., 2019; Williams and Zipser, 1989), regardless of the specific task.

5 Empirical Evaluation

5.1 Baselines

We evaluate our models across various scenarios, contrasting them with both multilingual and Arabic sequence-to-sequence pretrained language models. Specifically, we make use of mT5 (Xue et al., 2020) and mT0 (Muennighoff et al., 2022) as multilingual pretrained models; while comparing to AraBART (Eddine et al., 2022) and AraT5$_{v1}$ (Nagoudi et al., 2022b) as Arabic models. We evaluate our AraT5$_{v2}$ models (under different settings) and the selected baseline models on all eight NLG tasks (i.e., labeled data) described in Section 3.2.

5.2 Experimental Setup

For our experiments, we have two settings: one for the pretrained models and another for models we finetuning. We now describe each of these settings.

5.2.1 Pretrained Models

To pretrain our *AraT5$_{v2}$* model from scratch, we use the unsupervised pertaining strategy described in Section 4.1. We pretrain for one million steps on a Google TPU POD `v3-128`.[12] We employ a constant learning rate of 1e^{-3} and a dropout rate of 0.1. We use a batch size of $1,024$ with sequence length $2,048$. We further pretrain AraT5$_{v2}$ incorporating both unsupervised and supervised data (i.e., *joint strategy*; see Section 4.3), with the same hyperparameters for an additional 200K steps. We refer to the resulting model as *AraT5$_{v2}$-joint*.

5.2.2 Single Task Finetuning

We finetune both AraT5$_{v2}$ and AraT5$_{v2}$-joint, as well as baseline models, on the eight NLG tasks (20 datasets) for 20 epochs. We use a learning rate of 5e^{-5}, a batch size of 8, and a maximum sequence length of 512.[13] In all single task experiments, we consistently select the best checkpoint for each model based on performance on the respective development set. Subsequently, we report performance of each model on the respective test set.

5.2.3 Multitask Finetuning

We extend the pretraining of *AraT5$_{v2}$* and *AraT5$_{v2}$-joint* with labeled data by an additional 100K steps for each model, all within the multitask finetuning setting. These experiments are conducted using a Google TPU POD `v3-128` with the same hyperparameters as the initial pretraining.[14] For model comparisons in the single task setting, we calculate the average of three runs of finetuned Arabic and multilingual models on the test sets of each task. However, for the joint and multitask models, we incorporate labeled data during the subsequent pretraining phase, employing a fixed number of steps—200K for the joint model and 100K for the multitask model. As a result, we conduct a single evaluation run for these models due to the high computation costs.

[12] https://sites.research.google/trc/about/

[13] For GEC, we use a maximum sequence length of $1,024$.

[14] *AraT5$_{v2}$-mTask* trains for a total of 1.1M steps, whereas *AraT5$_{v2}$-joint-mTask* undergoes training for 1.3M steps.

Task	Test Set	Metric	Baselines				AraT5$_{v2}$		AraT5$_{v2}$-Joint		
			mT0	mT5	AraBART	AraT5$_{v1}$†	sTask	mTask⋆	Joint⋆	sTask	mTask⋆
DIAC	ADT ↓	CER	1.58$^{\pm0.13}$	1.64$^{\pm0.11}$	23.43$^{\pm1.51}$	2.58$^{\pm0.19}$	**1.30$^{\pm0.20}$**	1.97	2.20	1.90$^{\pm0.24}$	1.74
GEC	QALB 2014	F$_{0.5}$ (M^2)	65.86$^{\pm0.67}$	66.45$^{\pm0.22}$	68.67$^{\pm0.08}$	64.92$^{\pm0.23}$	70.52$^{\pm0.15}$	62.36	62.36	**70.73$^{\pm0.27}$**	64.36
	QALB 2015 L1		66.90$^{\pm0.92}$	66.68$^{\pm0.08}$	69.31$^{\pm1.55}$	64.22$^{\pm0.82}$	70.8$^{\pm0.12}$	62.46	62.46	**71.17$^{\pm0.16}$**	64.93
	ZAEBUC		47.33$^{\pm3.34}$	46.90$^{\pm0.87}$	82.08$^{\pm7.54}$	75.78$^{\pm2.43}$	**85.52$^{\pm0.69}$**	37.89	42.25	84.87$^{\pm0.58}$	78.30
PARA	TAPACO	Belu	15.43$^{\pm0.64}$	14.89$^{\pm0.28}$	**17.90$^{\pm1.06}$**	15.90$^{\pm0.06}$	16.82$^{\pm0.41}$	11.73	10.39	18.14$^{\pm0.84}$	11.68
	APB		**38.36$^{\pm0.14}$**	24.29$^{\pm13.98}$	37.66$^{\pm1.01}$	20.34$^{\pm1.82}$	35.04$^{\pm0.89}$	19.57	16.92	36.89$^{\pm0.44}$	16.93
	SemEval		20.49$^{\pm0.13}$	20.23$^{\pm0.03}$	24.52$^{\pm0.62}$	19.33$^{\pm0.08}$	25.52$^{\pm0.58}$	**72.53**	68.57	27.02$^{\pm0.53}$	72.72
QA	ARCD$_{QA}$	F$_1$	53.24$^{\pm0.24}$	51.63$^{\pm1.01}$	50.26$^{\pm0.99}$	58.12$^{\pm0.16}$	61.72$^{\pm0.89}$	55.43	53.84	**62.49$^{\pm0.69}$**	54.81
	TyDiQA$_{QA}$		76.31$^{\pm0.09}$	74.99$^{\pm0.23}$	73.32$^{\pm1.21}$	39.55$^{\pm1.96}$	82.99$^{\pm0.47}$	72.37	71.72	**84.21$^{\pm0.47}$**	72.44
	XSQUAD$_{QA}$		54.55$^{\pm0.76}$	47.43$^{\pm0.91}$	47.33$^{\pm0.87}$	48.71$^{\pm0.5}$	57.79$^{\pm1.08}$	63.73	63.39	59.42$^{\pm0.72}$	**64.89**
	LMQA$_{QA}$		49.17$^{\pm0.34}$	45.13$^{\pm0.35}$	47.24$^{\pm0.13}$	51.95$^{\pm0.09}$	54.48$^{\pm0.12}$	47.50	46.63	**55.02$^{\pm0.26}$**	48.70
QG	ARCD$_{QG}$	Belu	17.73$^{\pm0.99}$	17.62$^{\pm2.1}$	22.79$^{\pm0.66}$	16.8$^{\pm1.32}$	**24.13$^{\pm0.20}$**	19.86	19.23	22.48$^{\pm1.30}$	21.54
	TyDiQA$_{QG}$		30.22$^{\pm0.91}$	31.0$^{\pm0.97}$	33.64$^{\pm0.13}$	22.09$^{\pm1.85}$	33.50$^{\pm0.75}$	25.37	24.50	**34.05$^{\pm0.34}$**	26.18
	XSQUAD$_{QG}$		10.04$^{\pm0.01}$	9.96$^{\pm0.03}$	10.27$^{\pm0.31}$	9.21$^{\pm0.09}$	10.98$^{\pm6.91}$	6.65	1.94	**11.50$^{\pm0.41}$**	7.30
	MLQA$_{QG}$		6.04$^{\pm0.08}$	6.00$^{\pm0.38}$	7.02$^{\pm0.09}$	6.12$^{\pm0.42}$	**7.56$^{\pm0.27}$**	3.96	3.25	7.28$^{\pm0.11}$	3.66
SUM	XLSum	Rouge$_L$	21.46$^{\pm0.54}$	20.64$^{\pm0.31}$	26.64$^{\pm0.04}$	22.71$^{\pm1.36}$	27.15$^{\pm0.09}$	63.59	52.25	28.12$^{\pm0.12}$	**65.66**
	CrossSum		21.00$^{\pm0.38}$	20.29$^{\pm0.01}$	25.89$^{\pm0.09}$	22.14$^{\pm1.53}$	26.57$^{\pm0.06}$	59.45	50.82	27.56$^{\pm0.06}$	**61.31**
	MarSum		23.00$^{\pm0.17}$	22.57$^{\pm0.21}$	26.49$^{\pm0.03}$	21.71$^{\pm0.39}$	26.64$^{\pm0.06}$	20.49	19.04	**26.81$^{\pm0.06}$**	20.78
	MassiveSum		25.57$^{\pm0.11}$	22.88$^{\pm0.12}$	30.0$^{\pm0.11}$	15.89$^{\pm0.4}$	23.00$^{\pm0.00}$	27.22	25.75	**27.69$^{\pm0.07}$**	26.97
	ANTCorp		90.29$^{\pm0.11}$	88.84$^{\pm0.91}$	90.0$^{\pm0.20}$	86.64$^{\pm0.22}$	90.94$^{\pm0.14}$	87.39	86.92	90.85$^{\pm0.12}$	88.22
TG	Arabic NTG	Bleu	19.03$^{\pm0.34}$	19.23$^{\pm0.01}$	22.75$^{\pm0.09}$	19.55$^{\pm0.16}$	22.13$^{\pm0.08}$	22.54	21.33	22.37$^{\pm0.06}$	**22.94**
	XLSum		6.50$^{\pm0.17}$	6.51$^{\pm0.11}$	8.98$^{\pm0.18}$	7.44$^{\pm0.11}$	9.59$^{\pm0.17}$	6.21	5.91	**9.82$^{\pm0.14}$**	6.11
TR	ANTAEC ↓	CER	19.21$^{\pm0.48}$	18.93$^{\pm0.30}$	18.29$^{\pm0.29}$	20.74$^{\pm0.17}$	**18.06$^{\pm0.21}$**	31.50	33.00	19.25$^{\pm0.06}$	31.66
	ATAR ↓	CER	16.79$^{\pm0.15}$	16.68$^{\pm0.22}$	17.70$^{\pm0.05}$	36.51$^{\pm1.53}$	14.96$^{\pm0.05}$	33.63	35.90	**14.70$^{\pm0.05}$**	33.19
	NETTrans	Belu	55.70$^{\pm0.18}$	55.02$^{\pm0.47}$	54.15$^{\pm0.75}$	51.89$^{\pm0.64}$	**58.33$^{\pm0.70}$**	43.69	42.65	57.81$^{\pm0.66}$	43.18
		H-Score ↑	37.01	35.42	39.86	34.59	41.90	41.41	38.73	42.56	**42.89**
		L-Score ↓	12.53	12.42	19.81	19.94	**11.44**	22.37	23.70	11.95	22.20

Table 2: Average of three runs of finetuned Arabic and multilingual models on OCTOPUS test. **L-Score**: refers to the macro-average scores of tasks where a lower score ↓ is better. **H-Score**: refers to the macro-average scores of tasks where a higher score ↑ is better. OCTOPUS task clusters taxonomy: (DIAC, Diacritization), (GEC, Grammatical Error Correction), (PARA, Paraphrase), (QA, Question Answering), (QG, Question Generation), (SUM, Summarization), (TG, News Title Generation), and (TR, Transliteration). †We refer to vanilla AraT5 (Nagoudi et al., 2022b) as AraT5$_{v1}$. ⋆For the *joint* and *multitask* models, we utilize the labeled data during the further pretraining phase. Consequently, we employ it only once, as opposed to the regular single fine-tuning, which involves three runs. **Bold and green:** best score in the individual task. **Bold and orange:** best average scores over all tasks.

5.3 Evaluation Metrics

We present the results of our models and the baseline models independently on each task of evaluated datasets, using the relevant metric. We employ Bleu score as an evaluation metric for paraphrase, question generation, title (i.e. headline news) generation, and sentence-level transliteration tasks. Additionally, we use Rouge$_L$, F$_1$, and F$_{0.5}$ (M^2) as evaluation metrics for summarization, question answering, and grammatical error correction, respectively. For diacritization and word-level transliteration datasets, we utilize the character error rate (CER) metric. We split the evaluation scores into “L-Score” where lower ↓ is better (e.g., CER) and “H-Score” where higher ↑ is better, i.e., Bleu, F$_1$, F$_{0.5}$, and Rouge$_L$.

5.4 Results

Table 2 shows that our proposed models, across different settings, outperform the baseline models in ∼ 90% of the individual test sets (18 out of 20). Notably, AraT5$_{v2}$ significantly outperforms the vanilla AraT5$_{v1}$ (Nagoudi et al., 2022b) by 7.3 and 8.58 points in terms of the macro-average scores for tasks where *higher* (↑) and *lower* (↓) score is better, respectively. Furthermore, AraT5$_{v2}$ markedly outpaces the second-ranked baseline model, AraBART, by an average of 2.04 (↑) and 8.45 (↓) in the macro-average scores.

Additionally, the AraT5$_{v2}$-joint single-task model achieves the highest score in 8 out of 20 (∼ 40%) for the individual tasks, followed by the AraT5$_{v2}$ models and the AraT5$_{v2}$-joint multitask model, each achieving the best score in 4 out of 20

Input text	الخيل والليل والبيداء تعرفني *** والسيف والرمح والقرطاس والقلم
Target	الخَيْلُ وَاللّيْلُ وَالبَيْداءُ تَعرِفُني *** وَالسّيفُ وَالرّمحُ والقرْطاسُ وَالقَلَمُ
Multitask model	الْخَيلِ وَاللَّيْل وَالْبِيلادْ تَعْرِفَني *** وَالسَّيْفُ وَالرُّمْحُ وَقِرْطَاسُ وَالْقَلمِ
Single task model	الخَيْلُ وَاللَّيْلُ وَالْبَيْدَاءُ تَعْرِفُنِي *** وَالسّيْفُ وَالرُّمْحُ وَالْقِرْطَاسُ وَالْقَلَمُ
Input text	إبراهيم بن كنيف النبهاني، شاعر إسلامي، اشتهر بأبيات له أولها تعز فإن الصبر بالحر أجمل *** وليس على ريب الزمان معول تناقلت كتب الأدب أبياته وهو من شعراء الحماسة.
Target	إبراهيمُ بن كُنَيْفٍ النَّبْهانِيُّ، شاعرٌ إسلاميٌّ، اشْتُهِرَ بِأَبْياتٍ لَهُ أَوَّلُها تَعزَ فَإِنَ الصَبْرَ بالحُرِ أَجْمَلُ *** وَلَيْسَ عَلَى رَيْبِ الزَمَانِ مُعَوَل تناقَلَتْ كُتُبُ الأدبِ أبياتَهُ وهُوَ مِنْ شُعَراءِ الحَماسَةِ.
Multitask model	إبراهيم بن كَنِيسٍ النَّبَهَانِيُّ، شَاعٌ إِسْلَامِيٌّ، أُشْتُهِرَ بِأَبْيَاتٍ لَهُ أَوَّلُهَا تَعَزّ فَإِنَّ الصَّبْ بِالْأَحْرِّ أَجْمَلُ *** وَلَيْسَ عَلَى رَيْبِ الزَّمَانِ مَعْوَلٌ تَنَاقَلَتْ كُتُبُ الأَدَبِ أَبْيَاتِهِ وَهُوَ مِنْ شُعَبَاءِ الْحُمَاسَةِ.
Single task model	إِبْرَاهِيمُ بْنُ كُنَيْفٍ النَّبْهَانِيُّ، شَاعِرٌ إِسْلَامِيٌّ، أُشْتُهِرَ بِأَبْيَاتٍ لَهُ أَوَّلُهَا تَعُزْ فَإِنَّ الصَّبْرَ بِالْحُرِّ أَجْمَلُ *** وَلَيْسَ عَلَى رَيْبِ الزَّمَانِ مُعَوَّلٌ تَنَاقَلَتْ كُتُبُ الأَدَبِ أَبْيَاتِهِ وَهُوَ مِنْ شُعَرَاءِ الْحِمَاسَةِ.

Table 3: Examples of negative task interference in the **diacritization task**, both in a single-task and multitask. **Color taxonomy**: "**blue**" refers to the original text, "**red**" denotes a word-level error, "**light red**" indicates a partial diacritization error on one more letter, and "**green**" signifies correctness. For single task, we use "*AraT5*$_{v2}$*-sTask*" whereas we use "*AraT5*$_{v2}$*-joint-mTask*" model as the multitask model.

(∼ 20%) tasks. It is also noteworthy that AraBART and mT0 each obtain the best score in only one task.

5.5 Discussion

Exploring different pretraining settings allows us to derive unique insights. Examples of insights that can be gleaned from Table 2 include:

Addressing open-domain problems. We observe that sequence-to-sequence models like T5 encounter challenges when tackling open-domain question-answering tasks. For example, the results on the MLQA dataset demonstrate notably low performance across all evaluated models.

Handling lengthy sentences. Multitasking proves effective in addressing challenges when working with long texts, such as paragraphs or documents. It significantly excels in tasks involving long sequences. For instance, paraphrasing text such as the SmEval dataset and abstractive summarization like ARCD and XLSum all include long sequences. Conversely, it does not lead to significant improvements in short-text paraphrasing, such as those at the sentence level in datasets like APPB and TAPACO.

Negative task inference. Notably, multitask training in our experiments has a negative impact on character-level tasks. For instance, we randomly select two examples from an Arabic poetry website[15], remove diacritics from the input text, and require both the AraT5$_{v2}$-joint multitask and AraT5$_{v2}$ single task models to diacritize these examples. As shown in Table 3, the multitask model alters the words themselves, while the single task model preserves the input words (i.e., it focuses solely on adding diacritization to the character sequences).

5.6 Performance Comparison

One of our primary objectives in developing a new version of AraT5 is to improve the time required for the finetuning process (i.e., convergence time). Therefore, we conduct a comparison between AraT5$_{v1}$ and AraT5$_{v2}$, as well as the baselines models in this respect. This allows us to analyze their computational efficiency and gain

[15] https://poetry.dctabudhabi.ae/

News Article	
	أكد النجم البرازيلي نيمار مهاجم نادي الهلال أن الدوري السعودي بات أكثر قوة من الدوري الفرنسي مذكراً الجميع بتجربته في الأخيرعندما انتقل إلى باريس سان جيرمان صيف ٢٠١٧. وأوضح نيمار خلال مؤتمر صحفي مقام في بارا البرازيلية لدى سؤاله عن الدوري السعودي: وأؤكد لك أن كرة القدم هي نفسها ، الكرة هي نفسها و يسجلون الأهداف و بالنظر إلى الأسماء فإن الدوري السعودي بات أقوى من الدوري الفرنسي. التدريبات هناك شديدة ونتعطش أنا وزملائي للفوز هناك بشكل كبير والتتويج مع الهلال. وأضاف: الجميع اعتقد أن الدوري السعودي ضعيف والأمر نفسه حدث معي عندما انتقلت إلى الدوري الفرنسي، حينها ظن الناس الأمر نفسه لكني لم أضرب في حياتي من قبل المدافعين أكثرمن هناك. وأبان حول الدوري السعودي: اللاعبون الذين يلعبون هناك يعلمون مدى صعوبة اللعب في الدوري السعودي وأنا متأكد أنه لن يكون أمرا سهلا الفوز بالمسابقة لأن الفرق عززت صفوفها بلاعبين جدد، وستكون بطولة ممتعة وشيقة جدا. وتلعب البرازيل أمام بوليفيا في بارا البرازيلية يوم السبت قبل أن تواجه بيرو يوم الأربعاء ضمن تصفيات كأس العالم لمنتخبات أميركا الجنوبية.
Title Generation	
Output	نيمار: أعرف ماذا يعني اللعب في الدوري السعودي نيمار: الدوري السعودي أقوى من الفرنسي نيمار: أعرف ماذا يعني الفوز بالمباريات في الدوري السعودي نيمار: أعرف ماذا يعني أن الدوري السعودي أقوى من الفرنسي نيمار: أعرف أن الدوري السعودي أقوى من الدوري الفرنسي
Question Answering	
Question no. 1	متي تقام مباراة بوليفيا و البرازيل؟
Output	السبت
Question no. 2	متي انتقل نيمار الي باريس سان جيرمان؟
Output	صيف ٢٠١٧
Question Generation	
Answer	تلعب البرازيل أمام بوليفيا في بارا البرازيلية يوم السبت
Output	من يقابل البرازيل في تصفيات كأس العالم ؟

Table 4: OCTOPUS output examples based on a randomly picked article from a news website. We prompt OCTOPUS to generate five potential titles, answers based on the questions, and questions for the provided answer.

insights into their convergence behavior. To quantify this, we measure the required average time for convergence (in hours) and the average number of epochs needed to achieve convergence based on model results on development datasets. For a fair comparison, we finetune all models for a maximum of 20 epochs across all tasks. Notably, the evaluation results carry on the average of three separate runs using three different seeds, thereby enhancing the robustness and reliability of our comparison.

Convergence time. In general, we observe that AraBART and AraT5$_{\text{v2}}$ need on average 12 and 13 epochs, respectively, till convergence compared to AraT5$_{\text{v1}}$, which needs an average of 16 epochs to achieve the best performance. So, we notice that AraBART requires only 2.9 hours to converge and achieve the optimal performance, while AraT5$_{\text{v2}}$, and AraT5$_{\text{v1}}$, need an average of 3.77 and 5.20 hours, respectively, to reach the best score. So, we observe that AraT5$_{\text{v2}}$ is approximately 35.19% faster than AraT5$_{\text{v1}}$ in terms of training times.[16]

6 OCTOPUS Toolkit

6.1 Model Selection

Our objective is to introduce a versatile language generation toolkit capable of handling a wide range of tasks, all within a single model. To achieve this goal, we have explored multiple training strategies, as described in Section 4. Based on our empirical evaluations, we observe that finetuned *AraT5$_{v2}$-joint* under the multitask setting outperforms the other models.[17] Therefore, we utilize this model as the foundation for developing our OCTOPUS toolkit (illustrated in Figure 1).

[16] We note that AraBART requires a shorter time to converge due to the vocabulary size of the model (32K) compared to (110K) for AraT5-based models.

[17] As Table 2 shows, *AraT5$_{v2}$-joint-mTask* outperforms other models where a higher score is better and remains highly competitive in scenarios where a lower score is preferred.

Paraphrase	
Input	بالتزامن مع استنفار ليبيا لاسيما في الشرق جراء الإعصار آدانيالْ الذي ضربها، وصلت العاصفة دانيال أيضا إلى مصر وضربت محافظتين في شمال البلاد، هما مرسى مطروح والإسكندرية فقد غطت العاصفة مناطق الحدود المصرية غرب البلاد، وشملت مدن السلوم وسيوة ومطروح والإسكندرية. وبينت القنواتالمختلفة لصور الأقمار الصناعية شكل العاصفة التي طرقت الأبواب المصرية. وأظهرت طبقات من السحب المختلفة والأتربة المصاحبة لها.
Output	بالتزامن مع حالة التأهب في شرق ليبيا بسبب إعصار دانيال، وصلت العاصفة إلى مصر لتغطي محافظتين شمال البلاد، وهما مرسى مطروح والإسكندرية. وأظهرت مختلف صور الأقمار الصناعية شكل العاصفة التي ضربت ليبيا، والتي غطت إلى حد كبير شرق مصر، ولا سيما في مدينتي سرت والسلوم، كما أظهرت مختلف طبقات الغيوم والغابات المصاحبة لها.
Summarization	
Input	اعترف نضال حسن، الذي يدافع عن نفسه، بقتل الجنود، متحججا بحماية المسلمين وعناصر طالبان في أفغانستان، ولكن القاضي العسكري رفض حجته «بحماية الآخرين». وإذا أدين حسن، البالغ من العمر ٤٢ عاما، بقتل ١٣ شخصا وجرح آخرين فإنه سيواجه عقوبة الإعدام. ويعتبر الحادث الأكثر دموية من بين الهجمات غير القتالية التي وقعت في قاعدة عسكرية أمريكية. وقال شهود عيان دخل في ٥ نوفمبر الثاني عام ٢٠٠٩ مصحة تعج بالجنود الذين كانوا ينتظرون أدوارهم إجراء فحوصات طبية أو التلقيح، ثم صعد على مكتب، وأطلق النار من سلاحين بيديه، دون توقف إلا لإعادة تعبئة السلاح. مواضيع قد تهمك نهاية وسيقدم ممثلو الادعاء أدلة تفيد بأن حسن مال إلى الأفكار المتطرفة، وكان يزور المواقع بحثا عن آّالجهاديين» وطالبان، ساعات قبل الهجوم. وكان الرائد حسن سيلتحق بالقوات الأمريكية في أفغانستان قبل أن ينفذ هجومه. «عنف في مكان العمل» وصنفت وزارة الدفاع الأمريكية الحادث باعتباره «عنفا في مكان العمل» بدلا من تصنيفه «عملا إرهابيا»، وهو ما أغضب عائلات الضحايا، حسب ما أفاد به مراسل بي بي سي، نك براينت، في فروت هود. ويتوقع أن يدلي العديد من جرحى الحادث بشاهاداتهم أمام المحكمة. وسيواجه حسن عددا من ضحاياه في قاعة المحكمة لأنه سيتولى الدفاع عن نفسه. وهو يستخدم كرسيا متحركا لأنه أصيب بالشلل، عندما أطلق عليه شرطي في القاعدة العسكرية النار.
Output	تنظر محكمة عسكرية أمريكية في وقت لاحق من اليوم في قضية الطبيب النفسي العسكري الأمريكي، نضال حسن، الذي اعترف بقتل ١٣ شخصا وإصابة أكثر من ثلاثين آخرين في إطلاق نار بقاعدة فورت هود منذ أربعة أعوام.
Grammatical Error Correction	
Input	لا زال كبير الشبيحه يظن ان ارواح وآلام الناس اقل كلفه من تخليه عن منصبه ، فلذلك اذا كان السوريون لا يرتضون بهذه المعادله المهينه ، فعليهم ان يهبوا هبه قويه واحده وياخذو حقوقهم من هذه العصابه عنوه ، اننا يا أحبائي ندفع ثمن اكثر من اربعين عام ومن الخنوع والذل والثمن سيكون غاليا ولكنه يستأهل هذه التضحيات
Output	لازال كبير الشبيحة يظن أن أرواح وآلام الناس أقل كلفة من تخليه عن منصبه ، فلذلك إذا كان السوريون لا يرتضون بهذه المعادلة المهينة ، فعليهم أن يهبوا هبة قوية واحدة ، ويأخذوا حقوقهم من هذه العصابة عنوة . إننا يا أحبائي ندفع ثمن أكثر من أربعين عام ، ومن الخنوع والذل ، والثمن سيكون غاليا ، ولكنه يستأهل هذه التضحيات .

Table 5: OCTOPUS output examples for *grammatical error correction*, *paraphrasing*, and *summarization*.

6.2 Task Coverage

OCTOPUS is designed for *eight* machine generation tasks, encompassing diacritization, grammatical error correction, news headlines generation, paraphrasing, question answering, question generation, and transliteration. This comprehensive package includes a Python library along with associated command-line scripts. Table 4 illustrates the output of OCTOPUS, generating five potential titles, answers derived from questions related to the content, and questions corresponding to a provided answer based on a randomly selected article from a news website. Moreover, Table 5 showcases examples of OCTOPUS for grammatical error correction, paraphrasing, and summarization. We now describe the intricacies of implementation and design of the OCTOPUS toolkit, along with its various configurable settings.

6.3 Implementation

We distribute OCTOPUS as a modular toolkit built using standard libraries including `PyTorch` (Paszke et al., 2019) and `HuggingFace` (Lhoest et al., 2021). It is implemented in Python and can be easily installed using the `pip` package. It is compatible with Python versions 3.8 and later, `Torch` version 2.0 and later, and the `HuggingFace Transformers` library version 4.30 or higher.[18] We offer three usage options with varieties of arguments: *(i) Command-Line Interface (CLI)*, *(ii) Python integration package*, and *(iii) an interactive web interface*.

CLI ommands. We offer three command-line interfaces for task selection and output generation as follows: First, the "*octopus_interactive*" command provides an interactive mode that allows users to actively engage with the system. With this command, users can efficiently select their desired task and input text and then apply the chosen task to generate output. For instance, if a user wants to diacritize several sentences, they can initiate the diacritization task and input the sentences one by one to undergo the diacritization process. Second,

[18] Installation instructions and documentation can be found at: https://github.com/UBC-NLP/octopus.

	Argument	**Description**
Basic	- - *help* [-*h*]	To display the arguments details
	- - *cache-dir* [-*c*]	Specify the path to the cache directory.
	- - *logging-file* [-*l*]	Define the file path for logging.
Task	- - *prefix* [-*p*]	Task prefix should be one of the following: ['*diacritize*', '*correct_grammar*', '*paraphrase*', '*answer_question*', '*generate_question*', '*summarize*','*generate_title*', '*translitrate_ar2en*', '*translitrate_en2ar*']
Input & Output	- - *text* [-*t*]	Provide the input text for generative tasks.
	- - *input-file* [-*f*]	Specify the path of the input file.
	- - *max-outputs* [-*o*]	Define the number of hypotheses to generate as output.
	- - *batch-size* [-bs]	Set the number of input sentences processed in a single iteration.
	- - *seq-length* [-*s*]	Specify the maximum sequence length for the generative text.
Decoding	- - *search-method* [-*m*]	Choose the decoding method from the options ['*greedy*', '*beam*', '*sampling*'].
	- - *nbeam* [-*nb*]	If using beam search, specify the beam search size.
	- - *no-repeat-ngram-size* [-*ng*]	Avoid repeating the same n-gram size in the generated text.
	- - *top-k* [-*k*]	Utilize sampling with a top-k strategy.
	- - *top-p* [-*p*]	Implement sampling with a top-p strategy.

Table 6: OCTOPUS command line argument list.

the main command "*octopus*" offers two options: users can either directly input the text or specify a file path, allowing flexibility in applying multiple tasks to a large amount of data points. Finally, the task-specific command "*octopus-taskname*" offers seven task-specific commands, each corresponding to one of the supported tasks. For instance, there are "*octopus-diacritize*" and "*octopus-paraphrase*" commands. These task-specific commands follow the same usage pattern as the "octopus" command, but are designed for individual tasks.

Python integration package OCTOPUS is a Python library that offers numerous functions for seamless integration with various dataframe architectures, including `Pandas`, `PySpark`, `Dask`, and more. It takes as input the function to be integrated into user code and returns both generative text and processing logs.

Interactive web interface. We offer a dynamic interactive web interface that allows users to try OCTOPUS tasks. Furthermore, to facilitate adoption, we provide a `Google Colab` notebook with detailed instructions on how to use the OCTOPUS tool and model, and integrate them with user's code.

6.4 Arguments

Each of the command lines (i.e., *octopus-interactive, octopus, or octopus-taskname* supports or requires several arguments. Furthermore, OCTOPUS supports four decoding methods on the decoder side: *greedy search*, *beam search* (Koehn, 2009), *top-k sampling* (Fan et al., 2018), and *nucleus sampling* (Holtzman et al., 2019). We set as the default setting *beam search* with a beam size of 5, and a maximum sequence length of 2, 048. Table 6 shows detailed descriptions of the arguments and their usage. This information helps users understand and utilize the provided arguments effectively.

7 Conclusion

We introduced a suite of powerful Arabic text-to-text Transformer models trained on large and diverse datasets, with an extended sequence length of up to 2, 048. We also explored various pretraining strategies, including unsupervised and joint pertaining, using both single and multitask settings. Our models outperform competitive baselines, demonstrating their effectiveness. Furthermore, we introduced OCTOPUS, a publicly available Python-based package and command-line toolkit tailored for *eight* Arabic natural language generation tasks. OCTOPUS is designed to be extensible, and we plan to expand its capabilities by adding more tasks and increasing the capacity of our back-end model.

8 Limitations

We identify the following limitations:

- **Dialectal Arabic**. In this paper, our primary focus is on MSA tasks. Nevertheless, we are committed to expanding our scope to cover tasks in available Arabic dialects in the future. Currently, there is a recognized necessity within the community to facilitate the creation of datasets tailored to multiple Arabic dialects. For example, there is currently a deficiency in dialectal resources for sequence-to-sequence tasks such as summarization, paraphrasing, and question-answering. As more resources are created for dialects covering these tasks, we anticipate enhancing the coverage and capabilities of OCTOPUS exploiting these resources. Fortunately, our toolkit and core back-end models are extensible and hence would allow for such a development seamlessly.

- **Task Coverage**. OCTOPUS currently encompasses only eight generation tasks. However, we have plans to expand its capabilities by including additional tasks. These upcoming additions can involve, for example, dialgoue geeration and tasks involving code-switching. Again, adding more tasks to OCTOPUS will not be onerous, once respective datasets are available.

- **Intended Use**. OCTOPUS is a natural language generation toolkit designed to handle eight different tasks. We have tried the toolkit under different scenarios and found it to perform well. However, before any real-world usecases, we strongly encourage further and more extensive evaluations under diverse conditions.

9 Ethical Considerations

Our pretraining datasets are sourced from the public domain. Similarly, the labeled datasets used for model finetuning have been collected from publicly available data, made possible through the dedicated efforts of numerous researchers over the years. Consequently, we do not have significant concerns regarding the retrieval of personal information from our trained models. It is essential to note that the datasets we gather to construct OCTOPUS may contain potentially harmful content. Furthermore, during model evaluation, there is a possibility of exposure to biases that could lead to unintended content generation. For release, all our pretrained models and the toolkit are publicly available for non-malicious use.

Acknowledgments

We acknowledge support from Canada Research Chairs (CRC), the Natural Sciences and Engineering Research Council of Canada (NSERC; RGPIN-2018-04267), the Social Sciences and Humanities Research Council of Canada (SSHRC; 435-2018-0576; 895-2020-1004; 895-2021-1008), Canadian Foundation for Innovation (CFI; 37771), Digital Research Alliance of Canada,[19] and UBC ARC-Sockeye.[20] We thank the Google TFRC program for providing us with free TPU access.[21]

References

Muhammad Abdul-Mageed, Chiyu Zhang, Azadeh Hashemi, and El Moatez Billah Nagoudi. 2019. AraNet: A Deep Learning Toolkit for Arabic Social Media. *arXiv preprint arXiv:1912.13072*.

Marwah Alian, Arafat Awajan, Ahmad Al-Hasan, and Raeda Akuzhia. 2019. Towards building arabic paraphrasing benchmark. In *Proceedings of the Second International conference on Data Science E-learning and Information Systems (DATA' 2019)*, pages 1–5.

Mohamed Seghir Hadj Ameur, Farid Meziane, and Ahmed Guessoum. 2019. Anetac: Arabic named entity transliteration and classification dataset. *arXiv preprint arXiv:1907.03110*.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637.

Abhik Bhattacharjee, Tahmid Hasan, Wasi Uddin Ahmad, Yuan-Fang Li, Yong bin Kang, and Rifat Shahriyar. 2021. Crosssum: Beyond english-centric crosslingual abstractive text summarization for 1500+ language pairs. *CoRR*, abs/2112.08804.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Amina Chouigui, Oussama Ben Khiroun, and Bilel Elayeb. 2021. An arabic multi-source news corpus: Experimenting on single-document extractive summarization. *Arabian Journal for Science and Engineering*, 46:3925–3938.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Kareem Darwish and Hamdy Mubarak. 2016. Farasa: A new fast and accurate Arabic word segmenter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1070–1074, Portorož, Slovenia. European Language Resources Association (ELRA).

[19] https://alliancecan.ca

[20] https://arc.ubc.ca/ubc-arc-sockeye

[21] https://sites.research.google/trc/about/

Moussa Kamal Eddine, Nadi Tomeh, Nizar Habash, Joseph Le Roux, and Michalis Vazirgiannis. 2022. Arabart: a pretrained arabic sequence-to-sequence model for abstractive summarization.

Ibrahim Abu El-Khair. 2016. 1.5 billion words arabic corpus. *arXiv preprint arXiv:1611.04033*.

Ali Fadel, Ibraheem Tuffaha, Bara' Al-Jawarneh, and Mahmoud Al-Ayyoub. 2019. Arabic text diacritization using deep neural networks.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Ibrahim Abu Farha and Walid Magdy. 2019. Mazajak: An online arabic sentiment analyser. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 192–198.

Kamel Gaanoun, Abdou Naira, Anass Allak, and Imade Benelallam. 2022. *Automatic Text Summarization for Moroccan Arabic Dialect Using an Artificial Intelligence Approach*, pages 158–177.

Nizar Habash and David Palfreyman. 2022. ZAEBUC: An annotated Arabic-English bilingual writer corpus. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 79–88, Marseille, France. European Language Resources Association.

Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Samin, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. Xl-sum: Large-scale multilingual abstractive summarization for 44 languages.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Nisarg Jhaveri, Manish Gupta, and Vasudeva Varma. 2019. clstk: The cross-lingual summarization toolkit. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 766–769.

Saad Khan, Jesse Hamer, and Tiago Almeida. 2021. Generate: A nlg system for educational content creation. In *EDM*.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Ivano Lauriola, Alberto Lavelli, and Fabio Aiolli. 2022. An introduction to deep learning in natural language processing: Models, techniques, and tools. *Neurocomputing*, 470:443–456.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019a. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, pages 7871–7880.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2019b. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Yuval Merhav and Stephen Ash. 2018. Design Challenges in Named Entity Transliteration. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 630–640, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Behrang Mohit, Alla Rozovskaya, Nizar Habash, Wajdi Zaghouani, and Ossama Obeid. 2014. The first QALB shared task on automatic text correction for Arabic. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 39–47, Doha, Qatar. Association for Computational Linguistics.

Hussein Mozannar, Karl El Hajal, Elie Maamary, and Hazem Hajj. 2019. Neural arabic question answering. *arXiv preprint arXiv:1906.05394*.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2022. Crosslingual generalization through multitask finetuning.

El Moatez Billah Nagoudi, Muhammad Abdul-Mageed, AbdelRahim Elmadany, Alcides Alcoba Inciarte, and Md Tawkat Islam Khondaker. 2022a. Jasmine: Arabic gpt models for few-shot learning. *arXiv preprint arXiv:2212.10755*.

El Moatez Billah Nagoudi, Ahmed El-Shangiti, AbdelRahim Elmadany, and Muhammad Abdul-Mageed. 2023. Dolphin: A challenging and diverse benchmark for arabic nlg. *arXiv preprint arXiv:2305.14989*.

El Moatez Billah Nagoudi, AbdelRahim Elmadany, and Muhammad Abdul-Mageed. 2022b. AraT5: Text-to-text transformers for Arabic language generation. In *Proceedings of the 60th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), pages 628–647, Dublin, Ireland. Association for Computational Linguistics.

El Moatez Billah Nagoudi, AbdelRahim Elmadany, and Muhammad Abdul-Mageed. 2022c. AraT5: Text-to-text transformers for Arabic language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 628–647, Dublin, Ireland. Association for Computational Linguistics.

El Moatez Billah Nagoudi, AbdelRahim Elmadany, and Muhammad Abdul-Mageed. 2022d. TURJUMAN: A public toolkit for neural Arabic machine translation. In *Proceedinsg of the 5th Workshop on Open-Source Arabic Corpora and Processing Tools with Shared Tasks on Qur'an QA and Fine-Grained Hate Speech Detection*, pages 1–11, Marseille, France. European Language Resources Association.

El Moatez Billah Nagoudi, AbdelRahim Elmadany, Muhammad Abdul-Mageed, Tariq Alhindi, and Hasan Cavusoglu. 2020. Machine generation and detection of arabic manipulated and fake news. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 69–84, Barcelona, Spain (Online). Association for Computational Linguistics.

Lorenz Nigst, Maxim Romanov, Sarah Bowen Savant, Masoumeh Seydi, and Peter Verkinderen. 2020. Openiti: a machine-readable corpus of islamicate texts. *http://doi. org/10.5281/zenodo*, 4075046.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. Camel tools: An open source python toolkit for arabic natural language processing.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1094–1101, Reykjavik, Iceland. European Language Resources Association (ELRA).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Alla Rozovskaya, Houda Bouamor, Nizar Habash, Wajdi Zaghouani, Ossama Obeid, and Behrang Mohit. 2015. The second QALB shared task on automatic text correction for Arabic. In *Proceedings of the Second Workshop on Arabic Natural Language Processing*, pages 26–35, Beijing, China. Association for Computational Linguistics.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Yves Scherrer. 2020. TaPaCo: A corpus of sentential paraphrases for 73 languages. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6868–6873, Marseille, France. European Language Resources Association (ELRA), European Language Resources Association.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures. In *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*. Leibniz-Institut für Deutsche Sprache.

Bashar Talafha, Analle Abuammar, and Mahmoud Al-Ayyoub. 2021. Atar: Attention-based lstm for arabizi transliteration. *International Journal of Electrical and Computer Engineering*, 11:2327–2334.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Daniel Varab and Natalie Schluter. 2021. MassiveSumm: a very large-scale, very multilingual, news summarisation dataset. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10150–10161, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Imad Zeroual, Dirk Goldhahn, Thomas Eckart, and Abdelhak Lakhouaja. 2019. Osian: Open source international arabic news corpus-preparation and integration into the clarin-infrastructure. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 175–182.