Actes de la 28e Conférence sur le Traitement Automatique des Langues Naturelles
Lille, France, 28 juin au 2 juillet 2021

Un modèle Transformer Génératif Pré-entraîné pour le _____ français

Antoine Simoulin[1, 2] Benoit Crabbé[2]
(1) Quantmetry, 8 rue d'Anjou, 75008 Paris, France
(2) Université de Paris, Olympe de Gouges, 8 Rue Albert Einstein, 75013 Paris
asimoulin@quantmetry.fr, benoit.crabbe@linguist.univ-paris-diderot.fr

RÉSUMÉ

Nous proposons une adaptation en français du fameux modèle *Generative Pre-trained Transformer* (GPT). Ce dernier appartient à la catégorie des architectures transformers qui ont significativement transformé les méthodes de traitement automatique du langage. Ces architectures sont en particulier pré-entraînées sur des tâches auto-supervisées et sont ainsi spécifiques pour une langue donnée. Si certaines sont disponibles en français, la plupart se déclinent avant tout en anglais. GPT est particulièrement efficace pour les tâches de génération de texte. Par ailleurs, il est possible de l'appliquer à de nombreux cas d'usages. Ses propriétés génératives singulières permettent de l'utiliser dans des conditions originales comme l'apprentissage sans exemple qui ne suppose aucune mise à jour des poids du modèle, ou modification de l'architecture.

ABSTRACT

Generative Pre-trained Transformer in _____ (French)

We introduce a French adaptation from the well-known GPT model. GPT relies on transformer pre-trained architectures, which profoundly transformed natural language processing methods. Such models are pre-trained using a self-supervised objective and are therefore specific to a given language. Although some models may exist in French, the majority is released in English only. GPT achieves impressive language generation performances. The model can address a large variety of tasks. In particular, the model may benefit from original configurations such as few-shot or zero-shot learning. In such configurations, it is possible to address tasks without any parameters fine-tuning.

MOTS-CLÉS : GPT, Génératif, Transformer, Pré-entraîné, français.

KEYWORDS: GPT, Transformer, Generative, Pre-trained, French.

1 Introduction

L'utilisation des modèles pré-entraînés et des architectures à base de transformers (Vaswani *et al.*, 2017) a significativement amélioré les performances des modèles de traitement automatique du langage (TAL). Les modèles GPT (Radford *et al.*, 2019a,b; Brown *et al.*, 2020) cherchent à améliorer le potentiel de génération automatique de texte en proposant un paradigme d'apprentissage et d'inférence qui permet d'utiliser le même modèle sur plusieurs tâches sans ajustement de son architecture. Cette particularité peut être étendue jusqu'à une configuration d'apprentissage sans exemple (en anglais, *zero-shot learning*). Dans cette configuration, les poids du modèle ne sont pas mis à jour sur des exemples spécifiques à une tâche.

Volume 1 : conférence principale, pages 246–255.

La généralité affichée par les modèles transformers a néanmoins ses limites. En premier lieu, le pré-entraînement est spécifique à la langue utilisée. L'adaptation de tels modèles dans d'autres langues est loin d'être trivial. Elle nécessite en particulier de très larges corpus — jusqu'à plusieurs milliards de *tokens*. L'entraînement des modèles requiert par ailleurs une importante puissance de calcul, cet étape est typiquement réalisée sur plusieurs dizaines de GPUs ou TPUs. Finalement, l'analyse de ces modèles et l'étude de leur pertinence suppose la disponibilité de *benchmarks* d'évaluation rigoureux.

A notre connaissance, aucun modèle transformer génératif n'a encore été proposé pour le français. Nous adaptons ainsi les modèles OpenAI GPT et GPT-2 (Radford *et al.*, 2019a,b). Nos contributions sont les suivantes :

— Nous proposons un corpus spécifique à l'entraînement des modèles de langues neuronaux en français. La construction du corpus est détaillée en SECTION 2.
— Nous avons entraîné deux modèles avec un très grand nombre de paramètres que nous diffusons en accès ouvert afin de favoriser leur étude et application dans un contexte académique comme industriel. L'architecture des modèles utilisés est décrite en SECTION 3.
— Nous proposons des *benchmarks* d'évaluations spécifiques aux modèles de langue, inspirés des *benchmarks* anglais pour favoriser la comparaison avec des modèles de langue alternatifs. L'ensemble des méthodes d'évaluation sont présentées en SECTION 4.

2 Construction des corpus

Constitution du corpus d'entraînement Les modèles transformers n'ont besoin que de texte brut pour le pré-entraînement. Néanmoins, le nombre important de paramètres nécessite l'utilisation de larges corpus. Dans le cas des modèles GPT, les documents utilisés sont plus longs que ceux utilisés pour des modèles tels que BERT (Devlin *et al.*, 2019). La plupart des corpus utilisés pour les modèles BERT en français : Camembert (Martin *et al.*, 2020) ou Flaubert (Le *et al.*, 2020a,b) s'appuient sur des documents relativement courts dont l'ordre n'était pas conservé. Nous avons donc dû préparer des corpus spécifiques. Nous avons agrégé deux corpus d'ordre de grandeurs différents pour entrainer les modèles. On résume leurs statistiques en TABLE 1.

Le premier corpus, utilisé pour l'entraînement de GPT$_{fr}$-124M, est une agrégation de corpus existants : Wikipédia, OpenSubtitle (Tiedemann, 2012) et Gutenberg. Les documents sont séparés en phrases. Les phrases successives sont ensuite concaténées dans la limite de 1 024 *tokens* par document.

Modèles	OpenAI GPT	OpenAI GPT-2	GPT$_{fr}$-124M	GPT$_{fr}$-1B
# Documents ($\times 10^6$)	2,26†	8,00	1,66	7,36
# *Tokens* ($\times 10^9$)	1,16†	4,68†	1,60	3,11
Moy. # *tokens* par document	512†	585†	965	422

TABLE 1 – Caractéristiques des corpus utilisés pour le pré-entraînement des modèles. Les données marquées † sont estimées à partir des caractéristiques disponibles. En particulier, pour le modèle OpenAI GPT on confond le nombre de *tokens* par document et la taille du contexte. Les caractéristiques du modèle OpenAI GPT-2 sont estimées à partir de l'échantillon proposé en accès ouvert.

Le second corpus est utilisé pour l'entraînement de notre modèle avec un milliard de paramètres : GPT$_{fr}$-1B. Il augmente le premier avec des données extraites du *Common Crawl* (Li *et al.*, 2019)

en français. Les données *Common Crawl* sont filtrées en plusieurs étapes en nous inspirant de la procédure proposée dans Brown *et al.* (2020). Dans un premier temps, nous avons exclu tous les documents courts, de moins de 128 *tokens* comme proposé dans Shoeybi *et al.* (2019). Ce filtre très simple a permis de filtrer plus de 93% des documents du corpus original. Nous avons ensuite filtré les documents dont la distribution des mots est très éloignée de celle de notre premier corpus. Pour cela nous avons entraîné un classifieur binaire à discriminer les documents issus de notre premier corpus et du *Common Crawl* en utilisant 200 000 documents tirés aléatoirement. Nous avons exclu tous les documents présentant une probabilité <10% d'être issu du corpus d'entraînement. Ce seuil volontairement assez large nous a permis d'écarter les documents explicitement mal formés ou incohérents. Nous avons finalement appliqué un second filtre selon la structure des documents. Nous avons sélectionné les documents présentant une perplexité [1] faible selon notre modèle GPT$_{fr}$-124M. Afin de conserver des documents hors de la distribution, on considère pour chacun un seuil g. Avec g une réalisation d'une loi de Pareto $G \sim \mathcal{G}(\alpha)$. Le document est conservé si sa perplexité ppl vérifie : $g > ppl/ppl_{seuil}$. Avec le seuil de perplexité ppl_{seuil} fixé à 60.

Corpus pour l'évaluation des modèles de langues Toujours dans une optique de s'aligner avec les travaux en anglais, nous avons constitué deux corpus d'évaluation d'un modèle de langue à partir de l'encyclopédie en ligne Wikipédia. Pour cela nous avons collecté le texte des articles en français, labélisés « articles de qualité » et « bons articles ». [2] Nous avons collecté le texte de 2 246 articles de qualité et 3 776 bons articles sur une période de 2003 à 2020. Nous n'avons pas appliqué de pré-traitements spécifiques. En effet, l'utilisation des modèles transformers suppose généralement d'utiliser une tokenization spécifique qui assure que très peu de *tokens* sont en dehors du vocabulaire. Les caractéristiques des corpus et de leurs homologues anglais sont présenté dans la TABLE 2. Nous précisons que **ces articles ont été spécifiquement exclus du corpus utilisé pour le pré-entraînement de nos modèles**.

Nous avons divisé aléatoirement les articles de qualité en des corpus d'entraînement/validation/test contenant respectivement 2 126/60/60 articles pour constituer le corpus **WikiText-2-FR**. Pour le corpus **WikiText-72-FR**, nous avons conservé les corpus de validation et de test inchangés. Le jeu d'entraînement correspond à la concaténation du jeu d'entraînement **WikiText-35-FR** et de l'ensemble des bons articles.

	WikiText-EN				WikiText-FR			
	Valid	Test	Train-2	Train-103	Valid	Test	Train-35	Train-72
Documents	60	60	600	28 475	60	60	2 126	5 902
Tokens ($\times 10^3$)	218	246	2 089	103 227	896	897	35 166	72 961
Vocabulaire			33 278	267 735			137 589	205 403
Hors du vocabulaire			2,6%	0,4%			0,8%	1,2%

TABLE 2 – Statistiques descriptives des corpus **WikiText-FR**. La taille du vocabulaire est évaluée en utilisant le tokenizer MOSES (Koehn *et al.*, 2007). La proportion de mots hors du vocabulaire correspond au nombre de *tokens* apparaissant moins de trois fois.

1. Etant donné une séquence $U = \{u_1 \cdots u_n\}$, on défini sa perplexité : $PPL(U) = exp\left(-\frac{1}{T}\sum_{i=1}^{T} logp_\theta(u_i|u_{<i})\right)$ avec $logp_\theta(u_i|u_{<i})$ la log-vraisemblance conditionnelle, selon notre modèle, du ith token sachant les *tokens* précédents $u_{<i}$.

2. L'extraction du texte d'articles Wikipédia n'est pas une opération triviale. Afin de nous assurer de la bonne qualité des documents, nous avons directement utilisé l'API Wikipédia pour extraire les articles pour ce corpus.

3 Modèles

Architecture du modèle Nous avons utilisé l'architecture proposée pour les modèles anglais OpenAI GPT [3]. Le modèle est pré-entraîné selon une tâche de modèle de langue. Étant donné un corpus d'entraînement décrit comme une séquence de *tokens* $U = \{u_1 \cdots u_n\}$, on optimise les paramètres Θ du modèle pour maximiser la log-probabilité suivante : $\mathcal{L}(U) = \sum_i logP\left(u_i|u_{i-k} \cdots u_{i-1}; \Theta\right)$. Avec k la taille de la fenêtre de contexte. L'architecture de GPT s'appuie sur des transformers et présente beaucoup de similarités avec BERT. Il s'agit de plusieurs couches successives de décodeurs telles que définies dans l'architecture transformers. À la différence de BERT, le modèle applique une attention multi-têtes uniquement sur les *tokens* qui précèdent la position considérée.

Configuration du modèle à l'inférence Une fois le modèle pré-entraîné, il est possible de l'utiliser comme les modèles transformers classiques. On ajoute ainsi une couche spécifique à la tâche en sortie du modèle. On ajuste ensuite l'ensemble des paramètres de manière incrémentale (en anglais, *fine-tuning*) en fonction des exemples $x_1 \cdots x_m$ et des labels correspondants y.

Il est également possible de formaliser les tâches pour tirer parti des propriétés génératives du modèle. On transforme alors le jeu de données en séquences $x_1 \cdots x_m[SEP]y$. Chaque tâche est ainsi formalisée comme un modèle de langue : le modèle doit "générer" le label y comme la suite de la séquence $x_1 \cdots x_m[SEP]$. Il n'est alors pas nécessaire de modifier l'architecture du modèle. Il est également possible de résoudre la tâche sans mise à jour des poids du modèle (apprentissage sans exemple). Cette configuration est illustrée pour le résumé automatique en SECTION 4.

Architectures Nous avons entraîné deux modèles, dont l'un avec plus d'un milliard de paramètres, détaillés en TABLE 3. En nous appuyant sur les travaux de Shoeybi *et al.* (2019), qui comparent de nombreuses configurations, nous avons proposé une architecture permettant de ne pas utiliser de parallélisation du modèle. En effet la répartition du modèle en module dispersés sur plusieurs unités de calcul est un facteur de ralentissement important pour l'entraînement. Finalement les modèles utilisent un vocabulaire de type bytepair encoding (BPE) avec 50 000 unités (Sennrich *et al.*, 2016) entraîné sur l'ensemble du corpus utilisé pour l'entraînement de GPT$_{fr}$-124M.

Infrastructures L'entraînement de GPT$_{fr}$-124M a été effectué sur un TPU v2-8 à partir de l'interface Google Colab. L'entraînement du modèle Fr GPT$_{fr}$-1B a été mené en utilisant le supercalculateur français Jean Zay. Un cumul de 140 heures de calcul a été effectué sur du matériel de type Tesla V100 (TDP de 300W). L'entraînement a été distribué sur 4 noeuds de calcul de 8 GPUs. Nous avons utilisé de la parallélisation de données afin de diviser chaque micro batch sur les unités de calcul. Les émissions totales sont estimées à 580.61 kgCO$_2$eq. [4]

Entraînement Nous avons gardé le même paramétrage pour les deux modèles. Le taux d'apprentissage est fixé à $1.5e^{-4}$ avec une phase d'échauffement (en anglais, *warm up*) de 2 000 itérations puis une décroissance (en anglais, *decay*) selon une fonction cosinus. Nous avons effectué 125 000

3. Nous nous sommes en particulier appuyés sur la librairie Transformers.
4. Les estimations ont été réalisées à l'aide du Machine Learning Impact calculator présenté dans (Lacoste *et al.*, 2019).

itérations en utilisant une taille de batch de 128 documents et la précision mixte (Micikevicius *et al.*, 2018). Les autres paramètres (initialisation, dropout ...) sont fixés selon Radford *et al.* (2019a).

Modèles	OpenAI GPT	OpenAI GPT-2	GPT$_{fr}$-124M	GPT$_{fr}$-1B
Taille du contexte	512	1 024	1 024	1 024
# Couches	12	48	12	24
# Têtes d'attentions	12	25	12	14
Dimension des *embeddings*	768	1 600	768	1 792
# Paramètres ($\times 10^6$)	117	1 558	124	1 017

TABLE 3 – Caractéristiques des architectures et comparaison avec les modèles OpenAI (Radford *et al.*, 2019a,b).

4 Evaluation

Modèle de langue Le premier intérêt du modèle est de générer du texte cohérent. Pour évaluer la perplexité de nos modèles, nous avons utilisé les corpus présentés en SECTION 2. Le corpus utilisé pour le pré-entraînement étant proche de celui utilisé pour l'évaluation, nous n'avons pas effectué d'entraînement supplémentaire. Nous présentons les résultats sur le jeu d'évaluation en TABLE 4. Nous précisons que nous évaluons la perplexité sur la base de la tokenization définie par le modèle. Cette dernière est identique pour les modèles GPT$_{fr}$-124M et 1B mais peut être différente pour d'autres modèles. En particulier, nous avons considéré un modèle de langue de type 5-grams avec lissage de kneser-ney (Ney *et al.*, 1994) en utilisant l'outil SRILM (Stolcke, 2002) comme référence.

Les approches ne sont pas directement comparables car la tokenization est différente et notre modèle est entraîné sur un volume de données beaucoup plus conséquent. Les résultats en TABLE 4 sont donc donnés à titre illustratif mais soulignent la performance de notre modèle GPT$_{fr}$-1B.

Modèles	Modèle 5-grams	GPT$_{fr}$-124M	GPT$_{fr}$-1B
WikiText-35-FR (ppl)	166,7	109,2	12,9
WikiText-72-FR (ppl)	99,1		

TABLE 4 – Perplexité de nos modèles. Nous n'avons pas mis à jour les modèles sur le jeu d'entraînement et la perplexité est directement mesurée sur le jeu de test qui sont identiques pour deux *benchmarks*. Le modèle n-gram est entraîné sur les corpus d'entraînements correspondants.

Résumé automatique Cette tâche permet d'exploiter les propriétés génératives du modèle. Nous utilisons la configuration proposée dans (Radford *et al.*, 2019a) qui permet d'utiliser le modèle sans ajuster son architecture. Nous rajoutons simplement le motif *"Pour résumer :"*[5] après le texte original pour encourager le modèle à générer un texte qui résume les articles. Les poids du modèle ne sont donc pas mis à jour et aucune donnée d'entraînement n'est utilisée.

5. Dans le modèle OpenAI GPT-2, le motif rajouté est "TL ;DR :" qui correspond à "Too Long ; Didn't Read." et qui est utilisé sur le forum Reddit comme marqueur pour résumer une discussion.

Nous avons considéré le jeu de données OrangeSum pour le résumé abstratif (Eddine *et al.*, 2020). Nous générons la suite du texte en utilisant la stratégie de top-k *random sampling* (Fan *et al.*, 2018) avec $k = 2$. On retient les 3 premières phrases parmi les 100 *tokens* générés. Nous comparons notre modèle à la référence qui consiste à considérer la première phrase du texte comme résumé. Nous comparons les métriques ROUGES [6] (Lin, 2004) en TABLE 5. Les métriques montrent que, dans cette configuration complexe, nos modèles parviennent tout juste à s'approcher de la référence proposée.

	Synthèse			Titre		
	R1	R2	RL	R1	R2	RL
Première phrase	22,1	7,1	15,3	18,6	7,7	15,0
GPT$_{fr}$-124M	17,5	3,1	12,1	13,9	2,3	9,7
GPT$_{fr}$-1B	16,6	3,4	11,5	10,2	2,6	8,4

TABLE 5 – Comparaison des résumés générés avec le titre de l'article ou la synthèse proposée. Nous utilisons le score ROUGE et le corpus OrangeSum (Eddine *et al.*, 2020). Nos modèles sont utilisés en apprentissage sans exemple et donc sans mise à jour des paramètres sur le jeu d'entraînement.

Nous avons analysé manuellement des exemples. Le texte généré est correct au niveau de l'ortographe et de la syntaxe. Il s'inscrit également dans le thème et dans la continuité des articles proposés. Néanmoins le texte généré se concentre généralement sur un détail spécifique de l'article pour venir ensuite l'étoffer en inventant parfois des éléments (Kryscinski *et al.*, 2019). La méthode, si elle permet de générer du texte cohérent, ne parvient pas à synthétiser complètement l'idée générale du texte.

***Benchmark* FLUE** Les modèles génératifs élargissent certaines perspectives des modèles de type BERT. Néanmoins, ce type de pré-entraînement ne permet pas d'atteindre les mêmes performances qu'avec un modèle prenant en compte l'ensemble du contexte. Lorsque que l'on compare directement les modèles anglais sur le *benchmark* GLUE, on observe une différence moyenne de plus de 4 points entre OpenAI GPT et BERT-base (Radford *et al.*, 2019a). Nous avons tout de même comparé notre modèle sur le *benchmark* FLUE en français en TABLE 6.

Modèles	Livres	CLS DVD	Musique	PAWS-X	XNLI	Moy.
mBERT†	86,2	86,9	86,7	89,3	76,9	85,2
CamemBERT†	92,3	93,0	94,9	90,1	81,2	90,3
FlauBERT-base†	93,1	92,5	94,1	89,5	80,6	90,0
FlauBERT-large†	95,0	94,1	95,9	89,3	83,4	91,5
GPT$_{fr}$-124M	88,3	86,9	89,3	83,3	75,6	84,7
GPT$_{fr}$-1B	91,6	91,4	92,6	86,3	77,9	88,0

TABLE 6 – Scores de précisions pour les tâches discriminatives du *benchmark* FLUE. Le symbole † désigne les scores rapportés de Le *et al.* (2020b,a).

6. Les métriques ROUGES sont une collection de métriques permettant de comparer des résumés automatiques avec un texte de référence en calculant la proportion de "n-grams" communs entre les deux textes. ROUGE-1 (R1) correpond aux uni-grams, ROUGE-2 (R2) aux bi-gram et ROUGE-L (RL) à la plus longue séquence commune de n-gram rapportée au nombre d'uni-gram de la référence.

Nous avons considéré les tâches suivantes :

— CLS est un jeu de données composé d'avis sur Amazon à classer comme positifs ou négatifs. Il contient 3 catégories de produits : livres, DVD et musique. Chaque cartégorie est divisée en 2 000 exemples d'entraînements, de validation et d'évaluation.

— PAWS-X contient des paires de phrases. Il s'agit d'une tâche de classification binaire pour identifier les paires dont les deux phrases sont sémantiquement équivalentes. Il y a 49 401 exemples pour l'entraînement, 1 992 pour la validation et 1 985 pour l'évaluation.

— XNLI contient des paires de phrases. La tâche consiste à prédire si la première (prémisse) implique la seconde (hypothèse). 392 702 paires sont utilisées pour l'entraînement, 2 490 pour la validation et 5 010 pour l'évaluation.

Cette fois-ci les poids de notre modèle sont mis à jour. Les hyper-paramètres sont fixés selon les recommandations de Le *et al.* (2020a,b). Comme attendu, les performances du modèle ne se hissent pas à celles obtenues avec des modèles de type BERT.

Utilisation sans apprentissage Le modèle GPT-3 (Brown *et al.*, 2020) peut être adapté pour de nombreux cas d'usages, simplement en décrivant la consigne de la tâche suivie par un certain nombre d'exemples (en anglais, *zero-shot* et *few-shots learning*). Cette méthode cherche à conditionner le comportement du modèle en formatant le texte proposé en entrée en fonction de la tâche à réaliser. Les résultats sont surprenants mais les mécanismes sous-jacents restent largement à explorer. Néanmoins, il semblerait que le nombre de paramètres soit l'un des facteurs clés pour le fonctionnement de cette méthode. Notre modèle semble ainsi moins performant que GPT-3 sur des questions de culture générale ou de logique. Par exemple, quand on soumet le texte suivant : "Si Jérôme est plus grand que Michel, qui est le plus petit ?" le modèle GPT$_{fr}$-1B génère "Michel" mais ce résultat nous a semblé difficile à reproduire pour des expériences analogues. Si l'on cherche à générer la suite de la phrase suivante, "Quatre plus quatre font" le modèle va générer "quatre" alors que GPT-3 obtient la bonne réponse pour des expériences similaires.

5 Conclusion et travaux futurs

Nous proposons une version française du modèle GPT. S'il n'égalise pas les performances brutes de BERT, ses propriétés génératives permettent de l'utiliser dans des configurations remarquablement flexibles. Comme illustré dans nos expérimentations pour le résumé automatique, l'utilisation d'une configuration sans apprentissage reste très difficile pour le modèle. Cette configuration ouvre néanmoins des perspectives différentes d'un apprentissage traditionnel. Finalement, nous espérons que les performances de modèles de langues obtenues favoriseront son utilisation pour des cas d'usages correspondants. À cet effet, l'ensemble de nos contributions, incluant les modèles et les données sont disponibles en accès ouvert.

Remerciements

Ces travaux ont bénéficié d'un accès aux moyens de calcul de l'IDRIS au travers de l'allocation de ressources 2020-AD011011823 attribuée par GENCI.

Références

BROWN T. B., MANN B., RYDER N., SUBBIAH M., KAPLAN J., DHARIWAL P., NEELAKANTAN A., SHYAM P., SASTRY G., ASKELL A., AGARWAL S., HERBERT-VOSS A., KRUEGER G., HENIGHAN T., CHILD R., RAMESH A., ZIEGLER D. M., WU J., WINTER C., HESSE C., CHEN M., SIGLER E., LITWIN M., GRAY S., CHESS B., CLARK J., BERNER C., MCCANDLISH S., RADFORD A., SUTSKEVER I. & AMODEI D. (2020). Language models are few-shot learners. In H. LAROCHELLE, M. RANZATO, R. HADSELL, M. BALCAN & H. LIN, Éds., *Advances in Neural Information Processing Systems 33 : Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

DEVLIN J., CHANG M., LEE K. & TOUTANOVA K. (2019). BERT : pre-training of deep bidirectional transformers for language understanding. In J. BURSTEIN, C. DORAN & T. SOLORIO, Éds., *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, p. 4171–4186 : Association for Computational Linguistics. DOI : 10.18653/v1/n19-1423.

EDDINE M. K., TIXIER A. J. & VAZIRGIANNIS M. (2020). Barthez : a skilled pretrained french sequence-to-sequence model.

FAN A., LEWIS M. & DAUPHIN Y. N. (2018). Hierarchical neural story generation. In I. GUREVYCH & Y. MIYAO, Éds., *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1 : Long Papers*, p. 889–898 : Association for Computational Linguistics. DOI : 10.18653/v1/P18-1082.

KOEHN P., HOANG H., BIRCH A., CALLISON-BURCH C., FEDERICO M., BERTOLDI N., COWAN B., SHEN W., MORAN C., ZENS R., DYER C., BOJAR O., CONSTANTIN A. & HERBST E. (2007). Moses : Open source toolkit for statistical machine translation. In J. A. CARROLL, A. VAN DEN BOSCH & A. ZAENEN, Éds., *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic* : The Association for Computational Linguistics.

KRYSCINSKI W., KESKAR N. S., MCCANN B., XIONG C. & SOCHER R. (2019). Neural text summarization : A critical evaluation. In K. INUI, J. JIANG, V. NG & X. WAN, Éds., *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, p. 540–551 : Association for Computational Linguistics. DOI : 10.18653/v1/D19-1051.

LACOSTE A., LUCCIONI A., SCHMIDT V. & DANDRES T. (2019). Quantifying the carbon emissions of machine learning. *CoRR*, **abs/1910.09700**.

LE H., VIAL L., FREJ J., SEGONNE V., COAVOUX M., LECOUTEUX B., ALLAUZEN A., CRABBÉ B., BESACIER L. & SCHWAB D. (2020a). Flaubert : des modèles de langue contextualisés pré-entraînés pour le français (flaubert : Unsupervised language model pre-training for french). In C. BENZITOUN, C. BRAUD, L. HUBER, D. LANGLOIS, S. OUNI, S. POGODALLA & S. SCHNEIDER, Éds., *Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). Volume 2 : Traitement Automatique des Langues Naturelle, Nancy, France, June 8-19, 2020*, p. 268–278 : ATALA et AFCP.

LE H., VIAL L., FREJ J., SEGONNE V., COAVOUX M., LECOUTEUX B., ALLAUZEN A., CRABBÉ B., BESACIER L. & SCHWAB D. (2020b). Flaubert : Unsupervised language model pre-training for french. In N. CALZOLARI, F. BÉCHET, P. BLACHE, K. CHOUKRI, C. CIERI, T. DECLERCK, S. GOGGI, H. ISAHARA, B. MAEGAARD, J. MARIANI, H. MAZO, A. MORENO, J. ODIJK & S. PIPERIDIS, Éds., *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, p. 2479–2490 : European Language Resources Association.

LI X., MICHEL P., ANASTASOPOULOS A., BELINKOV Y., DURRANI N., FIRAT O., KOEHN P., NEUBIG G., PINO J. & SAJJAD H. (2019). Findings of the first shared task on machine translation robustness. In O. BOJAR, R. CHATTERJEE, C. FEDERMANN, M. FISHEL, Y. GRAHAM, B. HADDOW, M. HUCK, A. JIMENO-YEPES, P. KOEHN, A. MARTINS, C. MONZ, M. NEGRI, A. NÉVÉOL, M. L. NEVES, M. POST, M. TURCHI & K. VERSPOOR, Éds., *Proceedings of the Fourth Conference on Machine Translation, WMT 2019, Florence, Italy, August 1-2, 2019 - Volume 2 : Shared Task Papers, Day 1*, p. 91–102 : Association for Computational Linguistics. DOI : 10.18653/v1/w19-5303.

LIN C.-Y. (2004). Rouge : A package for automatic evaluation of summaries. In *Text summarization branches out*, p. 74–81.

MARTIN L., MÜLLER B., SUÁREZ P. J. O., DUPONT Y., ROMARY L., DE LA CLERGERIE É., SEDDAH D. & SAGOT B. (2020). Camembert : a tasty french language model. In D. JURAFSKY, J. CHAI, N. SCHLUTER & J. R. TETREAULT, Éds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, p. 7203–7219 : Association for Computational Linguistics.

MICIKEVICIUS P., NARANG S., ALBEN J., DIAMOS G. F., ELSEN E., GARCÍA D., GINSBURG B., HOUSTON M., KUCHAIEV O., VENKATESH G. & WU H. (2018). Mixed precision training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings* : OpenReview.net.

NEY H., ESSEN U. & KNESER R. (1994). On structuring probabilistic dependences in stochastic language modelling. *Comput. Speech Lang.*, **8**(1), 1–38. DOI : 10.1006/csla.1994.1001.

RADFORD A., NARASIMHAN K., SALIMANS T. & SUTSKEVER I. (2019a). Improving language understanding by generative pre-training. OpenAI Blog : Improving Language Understanding with Unsupervised Learning.

RADFORD A., WU J., CHILD R., LUAN D., AMODEI D. & SUTSKEVER I. (2019b). Language models are unsupervised multitask learners. OpenAI Blog : Better Language Models and Their Implications.

SENNRICH R., HADDOW B. & BIRCH A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1 : Long Papers* : The Association for Computer Linguistics. DOI : 10.18653/v1/p16-1162.

SHOEYBI M., PATWARY M., PURI R., LEGRESLEY P., CASPER J. & CATANZARO B. (2019). Megatron-lm : Training multi-billion parameter language models using model parallelism.

STOLCKE A. (2002). SRILM - an extensible language modeling toolkit. In J. H. L. HANSEN & B. L. PELLOM, Éds., *7th International Conference on Spoken Language Processing, ICSLP2002 - INTERSPEECH 2002, Denver, Colorado, USA, September 16-20, 2002* : ISCA.

TIEDEMANN J. (2012). Parallel data, tools and interfaces in OPUS. In N. CALZOLARI, K. CHOUKRI, T. DECLERCK, M. U. DOGAN, B. MAEGAARD, J. MARIANI, J. ODIJK & S. PIPERIDIS,

Éds., *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, p. 2214–2218 : European Language Resources Association (ELRA).

VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. & POLOSUKHIN I. (2017). Attention is all you need. In I. GUYON, U. VON LUXBURG, S. BENGIO, H. M. WALLACH, R. FERGUS, S. V. N. VISHWANATHAN & R. GARNETT, Éds., *Advances in Neural Information Processing Systems 30 : Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, p. 5998–6008.