Résumé automatique multi-documents guidé par une base de résumés similaires

Florian Baud, Alex Aussem

Actes de CORIA-TALN 2023
Paris, France, 5 au 9 juin 2023

Résumé automatique multi-documents guidé par une base de résumés similaires

Florian Baud[1,2] Alex Aussem[1]
(1) LIRIS UMR 5205 CNRS, 25 Avenue Pierre de Coubertin, 69622 Villeurbanne, France
(2) Visiativ, 26 Rue Benoit Bennier, 69260 Charbonnières-les-Bains, France
florian.baud@liris.cnrs.fr, alexandre.aussem@liris.cnrs.fr

RÉSUMÉ

Le résumé multi-documents est une tâche difficile en traitement automatique du langage, ayant pour objectif de résumer les informations de plusieurs documents. Cependant, les documents sources sont souvent insuffisants pour obtenir un résumé qualitatif. Nous proposons un modèle guidé par un système de recherche d'informations combiné avec une mémoire non paramétrique pour la génération de résumés. Ce modèle récupère des candidats pertinents dans une base de données, puis génère le résumé en prenant en compte les candidats avec un mécanisme de copie et les documents sources. Cette mémoire non paramétrique est implémentée avec la recherche approximative des plus proches voisins afin de faire des recherches dans de grandes bases de données. Notre méthode est évalué sur le jeu de données *MultiXScience* qui regroupe des articles scientifiques. Enfin, nous discutons de nos résultats et des orientations possibles pour de futurs travaux.

ABSTRACT

Non-Parametric Memory Guidance for Multi-Document Summarization

Multi-document summarization is a difficult task in natural language processing, aiming to summarize information from several documents. However, the source documents are often insufficient to obtain a qualitative summary. We propose a retriever-guided model combined with non-parametric memory for summary generation. This model retrieves relevant candidates from a database and then generates the summary considering the candidates with a copy mechanism and the source documents. The retriever is implemented with Approximate Nearest Neighbor Search (ANN) to search large databases. Our method is evaluated on the *MultiXScience* dataset which includes scientific articles. Finally, we discuss our results and possible directions for future work.

MOTS-CLÉS : Résumé multi-document, Augmentée par recherche, Guidage.

KEYWORDS: Multi-document summarization, Retrieval augmented, Guidance.

1 Introduction

Le résumé multi-documents automatique s'effectue à l'aide de deux méthodes : extractive (Wang *et al.*, 2020; Liu *et al.*, 2021) ou par abstraction (Jin *et al.*, 2020; Xiao *et al.*, 2022). Les méthodes dites extractives classent les phrases des documents sources afin d'obtenir un résumé. Ces méthodes réutilisent bien les informations importantes pour construire un résumé de qualité mais manquent de cohérence entre les phrases. Pour surmonter ce problème, les méthodes par abstraction sont étudiées pour rendre les résumés obtenus avec une meilleure cohérence. Les modèles générant des résumés

Actes de la 30e Conférence sur le Traitement Automatique des Langues Naturelles (TALN), volume 4 : articles déjà soumis ou acceptés en conférence internationale, pages 19–27.

par abstraction montrent d'excellentes performances sur le style d'écriture, mais oublient souvent des informations clés pour obtenir un résultat de qualité.

Pour que les modèles par abstraction tiennent compte des informations essentielles, (Dou *et al.*, 2021) guide leur modèle avec des informations supplémentaires comme un ensemble de mots-clés, des triplets de graphes, des phrases importantes des documents sources ou des résumés similaires récupérés depuis une base de connaissances. Avec comme mesure de similarité, la similarité cosinus. Leur méthode, qui utilise toutes les formes d'informations mentionnées précédemment, améliore la qualité et la contrôlabilité du résumé par rapport aux modèles non guidés. Cependant, le guidage nécessite des données d'entraînement spécifiques, notamment pour les mots-clés, les triplets de graphes et les phrases surlignées.

Notre proposition est qu'en guidant avec des résumés préexistants, le modèle puisse s'inspirer du résumé dans sa globalité mais aussi de pouvoir extraire des mots-clés et des phrases en utilisant un mécanisme de copie (Gu *et al.*, 2016; See *et al.*, 2017). Par conséquent, ce travail se concentre sur le guidage par des résumés similaires extraits d'une base de connaissances en utilisant la similarité cosinus. Le modèle, inspiré de RAG (Lewis *et al.*, 2020), est entièrement différentiable. En outre, le générateur du modèle utilise un mécanisme de copie sur les résumés remontés de la base de connaissances inspiré de (Cai *et al.*, 2021). Les conclusions de ces deux travaux ont motivé l'élaboration de notre modèle pour la tâche de résumé de texte multi-documents.

Nous démontrons le potentiel de notre méthode sur la base *MultiXScience* (Lu *et al.*, 2020) regroupant des articles scientifiques. Dans le cas d'articles scientifiques, les documents sources sont souvent insuffisants pour générer la partie *"related work"*. Des connaissances externes sont nécessaires pour rédiger un tel paragraphe. Notre objectif est de générer la partie *"related work"* avec notre méthode en intégrant des informations externes, dans notre cas des résumés préexistant et qui sont proches du résumé cible.

Dans cet article, nous étudions un modèle séquence à séquence guidé par une mémoire non paramétrique de résumés similaires. Notre contribution est double : premièrement, nous intégrons une mémoire non paramétrique comme définis dans (Lewis *et al.*, 2020) afin de récupérer les candidats à la génération du résumé, et deuxièmement, nous utilisons un mécanisme de copie pour intégrer ces candidats dans la procédure de génération. Le code de notre travail est disponible sur github [1].

2 Travaux similaires

Nous commençons par une revue des travaux similaires, tout d'abord les travaux sur les résumés d'articles scientifiques. En effet, (Cohan *et al.*, 2018) proposent de résumer des articles scientifiques provenant de *Arvix* et *Pubmed*, ils capturent la structure du document pour mieux représenter l'information du document source. Cependant cela concerne le résumé automatique d'un seul document. Dans le même but, (Cohan & Goharian, 2018; Yasunaga *et al.*, 2019) proposent de résumer un article avec les articles qui le citent. L'inconvénient de cette méthode est qu'elle ne peut pas être utilisée lors de la rédaction d'un article. Dans ce travail, nous utilisons les méthodes de résumé de texte multi-documents avec les références et non les articles qui citent les documents à résumer contrairement aux deux travaux précédents.

Les modèles utilisant des guidages sont proches de notre travail. (Cao *et al.*, 2018; Dou *et al.*, 2021)

1. https://github.com/florianbaud/retrieval-augmented-mds

extraient des résumés similaires d'une base de connaissances pour aider la génération du résumé cible. Cependant, ils utilisent des systèmes de recherche d'information tels que *ElasticSearch* pour trouver des candidats à la génération du résumé. De même, (An *et al.*, 2021) introduisent un système de recherche avec des vecteurs denses pour le résumé de texte, mais ils n'entraînent pas le système de recherche avec le reste du modèle. Dans notre cas, la recherche est effectuée avec des vecteurs denses et est entraînable pour trouver les candidats les plus pertinents à la génération du résumé.

Le domaine des modèles augmentés avec un système de recherche d'information différentiable partagent des points communs avec notre travail. Rag, (Lewis *et al.*, 2020) qui a introduit ce type de modèle, est utilisé pour la tâche de questions-réponses, où un contexte est donné pour répondre à la question. Le modèle récupère plusieurs contextes avec un système de recherche d'information entraînable end-to-end, puis répond à la question en utilisant chacun des candidats récupérés. Ces types de modèles sont également utilisés dans la tâche de traduction, où (Cai *et al.*, 2021) traduit une phrase avec une base de traduction préétablie. Leur modèle recherche dans cette base des traductions possibles de la phrase à traduire, puis les intègre dans la génération de la traduction par un mécanisme de copie. L'architecture que nous proposons repose sur la même idée : elle est basée sur un système de recherche d'information différentiable qui intègre la mémoire au moyen d'un mécanisme de copie. Il est intéressant de déterminer si le mécanisme de copie employé avec succès en traduction apporte également un gain au résumé multi-documents.

3 Méthode proposée

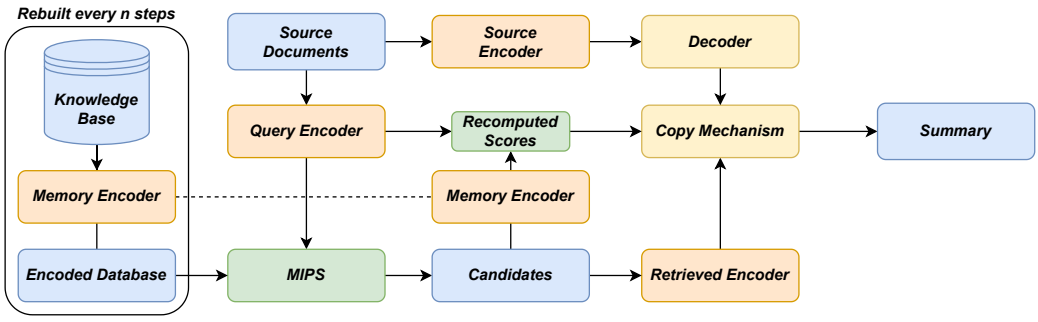

FIGURE 1 – Dans un premier temps, tous les éléments de la base de connaissances sont encodés avec un encodeur dédié : le *"Memory Encoder"*. Les documents sources sont transformés avec deux encodeurs distincts : le *"Query Encoder"* se charge d'encoder les documents sources afin de rechercher dans la base de connaissances et le *"Source Encoder"* se charge de représenter les documents sources pour la génération du résumé. Après avoir récupéré le top-k de la recherche, les candidats sont encodés avec l'encodeur *"Retrieved Encoder"* et avec le *"Memory Encoder"* pour recalculer le score de pertinence afin de propager le signal lors de la rétro-propagation du gradient. Finalement, le décodeur prend en entrée les documents sources et candidats pour la génération du résumé.

Inspiré par (Cai *et al.*, 2021), nous proposons un modèle composé d'un système de recherche d'information différentiable end-to-end et d'un générateur augmenté avec un mécanisme de copie. Nous commençons par présenter le système de recherche d'information différentiable puis le décodeur augmenté avec le mécanisme de copie. La figure 1 illustre l'architecture du modèle dans sa globalité.

3.1 Encodeur à mémoire non paramétrique

Les systèmes de recherche d'informations consistent en une requête et une base de recherche, l'objectif étant de retrouver les éléments pertinents de la base à l'aide de la requête. Dans notre cas les documents sources et les résumés cibles correspondent aux requêtes et à la base de recherche ou mémoire, respectivement noté Q et M. Ces documents sont encodés avec un modèle de type *Longformer* (Beltagy *et al.*, 2020). *LongFormer* possède une architecture *Transformer* (Vaswani *et al.*, 2017) qui peut traiter de longues séquences d'entrée avec une attention fenêtrée sur tous les tokens et une attention globale sur quelques tokens. Nous encodons les documents sources et les résumés avec deux encodeurs pré-entraînés distincts, un pour les documents sources et l'autre pour les résumés :

$$h^q = LED^q_{enc}(q) \quad q \in Q$$
$$h^m = LED^m_{enc}(m) \quad m \in M$$

où l'encodeur *LongFormer* est désigné par LED_{enc}. Tous les résumés de la base de connaissances sont encodés et stockés en amont de l'entraînement. Lors de la recherche dans la base, nous prenons le token *[CLS]* correspondant à un token spécial censé représenter le sens global du document encodé. Ce token est normalisé afin de calculer un score avec une fonction de pertinence :

$$h^q_{cls} = norm(h^q_{cls})$$
$$h^m_{cls} = norm(h^m_{cls})$$
$$score(x, y) = x^\top \cdot y$$

Le score représente la similarité cosinus entre les documents sources q et les documents de la mémoire m qui tombent dans l'intervalle $[-1, 1]$. Pour une recherche rapide, nous retrouvons les k documents les plus pertinents $m_{topk} = (m_1, \ldots, m_k)$ de la mémoire en utilisant la bibliothèque FAISS (Johnson *et al.*, 2021). À chaque propagation en avant de l'entraînement, les vecteurs des documents candidats $\{h^m_{cls,i}\}^k_{i=1}$ sont recalculés ainsi que les scores de pertinence $\{s_i = score(h^m_{cls,i}, h^q_{cls})\}^k_{i=1}$ afin de propager le signal jusqu'à l'encodeur de la mémoire comme dans (Cai *et al.*, 2021; Lewis *et al.*, 2020). De ce fait, les scores recalculés viennent biaiser le mécanisme de copie du décodeur permettant la propagation du signal jusqu'aux encodeurs.

L'encodeur de la mémoire ne ré-encode pas toute la base de connaissances à chaque étape de l'entraînement car cela serait un calcul coûteux. Au lieu de cela, la base de connaissances et l'index FAISS sont mis à jour à intervalles réguliers. D'autre part, nous encodons les candidats les plus recherchés (top-k) et les documents sources avec deux encodeurs distincts, LED^r_{enc} et LED^s_{enc}, comme indiqué ci-dessous :

$$h^r_{topk} = LED^r_{enc}(m_{topk})$$
$$h^s = LED^s_{enc}(q)$$

Le décodeur prend en compte ces deux résultats avec de l'attention croisée.

3.2 Le décodeur avec mécanisme de copie

Dans la partie décodeur de notre modèle, nous utilisons le décodeur de *LongFormer* et nous appliquons un mécanisme de copie aux candidats précédemment récupérés. Ainsi, nous avons :

$$h^d = LED_{dec}(y, h^s)$$

où LED_{dec} correspond à la partie décodeur du modèle *LongFormer*, et y est le résumé ciblé. Le décodeur prend en compte les documents sources h^s et les tokens précédents $y_{1:t-1}$, produisant un état caché h^d_t à chaque pas de la génération t. La probabilité du token suivant est calculée avec une fonction $softmax$:

$$P_{dec}(y_t) = softmax(W_d \cdot h^d_t + b_d) \tag{1}$$

où W_d est une matrice $hiddens_{size} \times vocab_{size}$ et b_d un vecteur de biais ; ces deux paramètres sont entraînés. Ensuite, nous incorporons les candidats du top-k m_{topk} avec un mécanisme de copie en calculant une attention croisée entre h^d_t et h^r_{topk}. Pour cela, nous réutilisons la partie attention croisée de *LongFormer* pour l'ajouter après son décodeur original. Cette nouvelle couche n'a qu'une seule tête d'attention afin d'utiliser les poids d'attention comme probabilité de copier un mot parmi les candidats du top-k. Étant donné k documents encodés dans h^r_{topk}, nous pouvons construire un ensemble de plongements (embeddings) de mots $\{r_{i,j}\}_{j=1}^{L_i}$ où $i \in [1, k]$, $j \in [1, L_i]$ et L_i est la longueur du document i. Concrètement, le poids d'attention du $j$ème token dans le $i$ème document pertinent est exprimé comme suit,

$$\alpha_{ij} = \frac{\exp(h^{d\top}_t W_a r_{i,j} + \beta s_i)}{\sum_{i=1}^{k}\sum_{j=1}^{L_i}\exp(h^{d\top}_t W_a r_{i,j} + \beta s_i)}$$

$$c_t = W_c \sum_{i=1}^{k}\sum_{j=1}^{L_i} \alpha_{ij} r_{i,j}$$

où W_a et W_c sont des paramètres entraînable, c_t est une représentation pondérée des k meilleurs candidats et β est un scalaire entraînable qui contrôle le score de pertinence entre les candidats récupérés et l'état caché du décodeur, permettant le flux de gradient vers les encodeurs comme dans (Cai *et al.*, 2021; Lewis *et al.*, 2020). L'équation 1 peut être réécrite pour inclure la mémoire :

$$P_{dec}(y_t) = softmax(W_d \cdot (h^d_t + c_t) + b_d) \tag{2}$$

Ainsi, la probabilité du prochain token prend en compte les poids d'attention des k candidats les plus importants. La probabilité finale du prochain token est donnée par :

$$P(y_t) = (1 - \lambda_t)P_{dec}(y_t) + \lambda_t \sum_{i=1}^{k}\sum_{j=1}^{L_i} \alpha_{ij} \mathbb{1}_{r_{ij}=y_t}$$

où λ_t est un scalaire agissant comme une porte logique calculé par un réseau feed-forward $\lambda_t = g(h^d, c_t)$. Le modèle est entraîné avec la fonction de perte de la log-vraisemblance $\mathcal{L} = -\log P(y^*)$ où y^* est le résumé cible.

3.3 Détails de l'entraînement

Notre modèle est composé de plusieurs encodeurs et décodeurs basés sur le *LongFormer* (Beltagy *et al.*, 2020). La taille de notre modèle est de 1.9 milliard de paramètres entraînables. Pour entraîner le modèle, nous avons utilisé la librairie *DeepSpeed* (Rasley *et al.*, 2020).

L'entraînement du modèle utilise les données *MultiXScience* comprenant 30 369 articles scientifiques pour l'entraînement, 5 066 articles de validation et 5 093 articles de test. L'objectif est de générer la partie *"related work"* en utilisant le résumé de l'article et les résumés des articles cités. Il s'agit d'un ensemble de données intéressant à expérimenter car l'écriture de la partie *"related work"* nécessite des connaissances extérieures aux documents sources.

Au début de l'apprentissage, les poids sont initialisés de façon aléatoire et l'encodeur sélectionne de "mauvais" candidats. Pour surmonter ce problème, nous pré-entraînons le système de recherche d'informations sur les données *MultiXScience* afin de commencer l'entraînement avec des résultats de bonne qualité. L'objectif est de maximiser la similarité entre le résumé et la section *"related work"*. Ces deux sections sont encodées avec les deux encodeurs précédents afin de calculer la similarité cosinus. Ainsi, pour une taille de données d'entraînement égale à N, nous avons N sections "abstract" encodées avec $A = \{LED^q_{enc}(a_i)\}^N_{i=1}$ et N sections "related work" encodées avec $B = \{LED^m_{enc}(b_j)\}^N_{j=1}$, le but est d'obtenir une similarité cosinus égal à 1 lorsque $j = i$ correspondant aux exemples positifs et -1 sinon pour les exemples négatifs. Nous calculons pour chaque élément de A, les erreurs suivantes :

$$\mathcal{L}_i(A, B) = -\log \frac{\exp\left(score(A_i, B_i)/\tau\right)}{\sum_{j=1}^{N} \exp\left(score(A_i, B_j)/\tau\right)}$$

où τ est un paramètre de température choisi arbitrairement. L'erreur finale est $\mathcal{L} = \sum_{i=1}^{N} \mathcal{L}_i$ rétro-propagée dans les deux encodeurs.

4 Expérimentation

Dans cette section, nous présentons les expériences réalisées sur le jeu de données *MultiXScience* pour évaluer notre modèle. L'entraînement du modèle complet est plus difficile en raison de sa taille mais aussi du problème du démarrage à froid du système d'informations. Ce dernier correspond au fait que les résumés similaires récupérés ne sont pas suffisamment pertinents pour aider le modèle à générer des résumés de qualité.

Pour réduire la charge de calcul, nous avons utilisé un modèle réduit où la base de connaissances n'est pas du tout reconstruite. En outre, les paramètres de l'encodeur de mémoire ont été gelés afin de réduire la complexité de l'apprentissage. Ces deux modifications ont permis de réduire considérablement le temps d'apprentissage. Le modèle réduit a moins de paramètres entraînables (1.4 milliard). Le modèle a été entraîné pendant deux jours sur quatre GPU v100 avec l'optimiseur Adam et un taux d'apprentissage de $1e-4$, une taille de données d'entraînement de 32, un top-k de 5 pour le récupérateur et avec 2k étapes de *warmup* et une décroissance linéaire jusqu'à 20k étapes. Les scores rouges (Lin, 2004) sur le jeu de données *MultiXScience* sont reportés dans le tableau 1.

Method	**R-1**	**R-2**	**R-L**
Notre modèle	28.9	6.2	17.6
(Xiao *et al.*, 2022)*	31.9	**7.4**	18.0
(Lu *et al.*, 2020)*	**33.9**	6.8	**18.2**

TABLE 1 – Le score ROUGE (R-1/R-2/R-L) de nos résultats sur le jeu de données de test MultiXScience. Le symbole * signifie que les résultats ont été empruntés à (Xiao *et al.*, 2022).

Malgré sa réduction de taille, nous observons que le modèle est compétitif par rapport à l'état de l'art avec des méthodes n'utilisant pas de mécanisme de copie couplé avec un système de recherche d'informations sur des résumés similaires.

5 Conclusion

Cet article présente une architecture pour le résumé de texte multi-documents inspirée par des modèles augmentés par un système de recherche d'informations. Cette architecture comprend un système de recherche d'informations qui cherche dans une base de connaissances des résumés similaires pour la génération d'un résumé. Ces documents sont intégrés dans la génération au moyen d'un mécanisme de copie. Une version réduite du modèle a été évaluée sur le jeu de données *MultiXScience*. Les résultats sont compétitifs par rapport à l'état de l'art, mais nous espérons améliorer encore nos résultats, d'une part en corrigeant correctement le problème du démarrage à froid, et d'autre part en utilisant le modèle complet. Pour la suite des travaux, nous prévoyons également d'augmenter la taille de la base de connaissances avec de nouvelles données, et d'appliquer notre méthode à d'autres jeux de données pour la tâche de résumé multi-documents.

Remerciements

Nous remercions le Centre de Calcul CNRS/IN2P3 (Lyon - France) pour la mise à disposition des ressources informatiques nécessaires à ce travail. De plus, ces travaux ont bénéficié d'un accès aux moyens de calcul de l'IDRIS au travers de l'allocation de ressources 2022-AD011013300 attribuée par GENCI. Enfin, nous remercions Roch Auburtin de la société Visiativ pour les conseils fournis sur nos travaux.

Références

AN C., ZHONG M., GENG Z., YANG J. & QIU X. (2021). Retrievalsum : A retrieval enhanced framework for abstractive summarization.

BELTAGY I., PETERS M. E. & COHAN A. (2020). Longformer : The long-document transformer. *arXiv :2004.05150*.

CAI D., WANG Y., LI H., LAM W. & LIU L. (2021). Neural machine translation with monolingual translation memory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, p. 7307–7318, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.acl-long.567.

CAO Z., LI W., LI S. & WEI F. (2018). Retrieve, rerank and rewrite : Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 152–161, Melbourne, Australia : Association for Computational Linguistics. DOI : 10.18653/v1/P18-1015.

COHAN A., DERNONCOURT F., KIM D. S., BUI T., KIM S., CHANG W. & GOHARIAN N. (2018). A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings*

of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 2 (Short Papers), p. 615–621, New Orleans, Louisiana : Association for Computational Linguistics. DOI : 10.18653/v1/N18-2097.

COHAN A. & GOHARIAN N. (2018). Scientific document summarization via citation contextualization and scientific discourse. *International Journal on Digital Libraries*, **19**(2), 287–303. DOI : 10.1007/s00799-017-0216-8.

DOU Z.-Y., LIU P., HAYASHI H., JIANG Z. & NEUBIG G. (2021). GSum : A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, p. 4830–4842, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.naacl-main.384.

GU J., LU Z., LI H. & LI V. O. (2016). Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 1631–1640, Berlin, Germany : Association for Computational Linguistics. DOI : 10.18653/v1/P16-1154.

JIN H., WANG T. & WAN X. (2020). Multi-granularity interaction network for extractive and abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, p. 6244–6254, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.acl-main.556.

JOHNSON J., DOUZE M. & JÉGOU H. (2021). Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, **7**(3), 535–547. DOI : 10.1109/TBDATA.2019.2921572.

LEWIS P., PEREZ E., PIKTUS A., PETRONI F., KARPUKHIN V., GOYAL N., KÜTTLER H., LEWIS M., YIH W.-T., ROCKTÄSCHEL T., RIEDEL S. & KIELA D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA : Curran Associates Inc.

LIN C.-Y. (2004). ROUGE : A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, p. 74–81, Barcelona, Spain : Association for Computational Linguistics.

LIU Y., ZHANG J., WAN Y., XIA C., HE L. & YU P. (2021). HETFORMER : Heterogeneous transformer with sparse attention for long-text extractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, p. 146–154, Online and Punta Cana, Dominican Republic : Association for Computational Linguistics. DOI : 10.18653/v1/2021.emnlp-main.13.

LU Y., DONG Y. & CHARLIN L. (2020). Multi-XScience : A large-scale dataset for extreme multi-document summarization of scientific articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 8068–8074, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.emnlp-main.648.

RASLEY J., RAJBHANDARI S., RUWASE O. & HE Y. (2020). Deepspeed : System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, p. 3505–3506, New York, NY, USA : Association for Computing Machinery. DOI : 10.1145/3394486.3406703.

SEE A., LIU P. J. & MANNING C. D. (2017). Get to the point : Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 1073–1083, Vancouver, Canada : Association for Computational Linguistics. DOI : 10.18653/v1/P17-1099.

VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. U. & POLOSUKHIN I. (2017). Attention is all you need. In I. GUYON, U. V. LUXBURG, S. BENGIO, H. WALLACH, R. FERGUS, S. VISHWANATHAN & R. GARNETT, Éds., *Advances in Neural Information Processing Systems*, volume 30 : Curran Associates, Inc.

WANG D., LIU P., ZHENG Y., QIU X. & HUANG X. (2020). Heterogeneous graph neural networks for extractive document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, p. 6209–6219, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.acl-main.553.

XIAO W., BELTAGY I., CARENINI G. & COHAN A. (2022). PRIMERA : Pyramid-based masked sentence pre-training for multi-document summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 5245–5263, Dublin, Ireland : Association for Computational Linguistics. DOI : 10.18653/v1/2022.acl-long.360.

YASUNAGA M., KASAI J., ZHANG R., FABBRI A. R., LI I., FRIEDMAN D. & RADEV D. R. (2019). Scisummnet : A large annotated corpus and content-impact models for scientific paper summarization with citation networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, **33**(01), 7386–7393. DOI : 10.1609/aaai.v33i01.33017386.