INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Unsupervised Dialogue Topic Segmentation in Hyperdimensional Space

Seongmin Park[1], *Jinkyu Seo*[2], *Jihwa Lee*[1]

[1]ActionPower, Seoul, Republic of Korea
[2]Dept. of Applied Biology and Chemistry, Seoul National University, Seoul, Republic of Korea
seongmin.park@actionpower.kr, seojk413@snu.ac.kr, jihwa.lee@actionpower.kr

Abstract

We present *HyperSeg*, a hyperdimensional computing (HDC) approach to unsupervised dialogue topic segmentation. HDC is a class of vector symbolic architectures that leverages the probabilistic orthogonality of randomly drawn vectors at extremely high dimensions (typically over $10,000$). HDC generates rich token representations through its low-cost initialization of many unrelated vectors. This is especially beneficial in topic segmentation, which often operates as a resource-constrained preprocessing step for downstream transcript understanding tasks. HyperSeg outperforms the current state-of-the-art in 4 out of 5 segmentation benchmarks – even when baselines are given partial access to the ground truth – and is 10 times faster on average. We show that HyperSeg also improves downstream summarization accuracy. With HyperSeg, we demonstrate the viability of HDC in a major language task. We open-source HyperSeg to provide a strong baseline for unsupervised topic segmentation.[1]

Index Terms: topic segmentation, hyperdimensional computing, summarization

1. Introduction

Automatic speech recognition (ASR) is often paired with downstream natural language processing (NLP) tasks for information extraction. Appropriately segmenting ASR transcripts to conform to the limited input length of downstream models is a practical and important consideration for such hybrid systems. Conventional NLP models cannot process whole meeting transcripts (which typically include hundreds or thousands of utterances spanning several hours) in a single inference context. Such physical constraints necessitate effective methods to divide transcripts into coherent segments before passing them on to downstream models.

Much research has focused on unsupervised approaches for topic segmentation due to the scarcity of labeled datasets with diverse domain and lexical coverage. Because segmentation is performed at the utterance level, dominant approaches to unsupervised topic segmentation use neural utterance embeddings. We find two problems with this line of research: neural utterance embeddings are very brittle to the domain in question [1, 2], and existing unsupervised topic segmentation methods cannot determine segment counts on their own. Even state-of-the-art (SOTA) systems rely on carefully tuned hyperparameters, without which the systems offer lower segmentation accuracy than random segmentation.

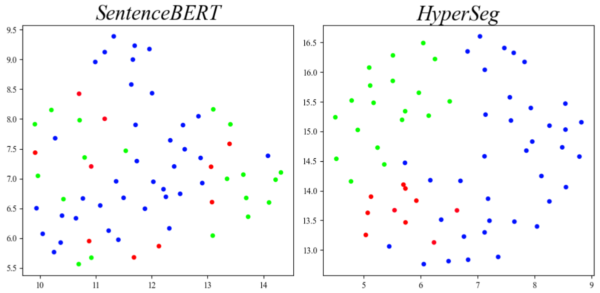

Figure 1: *Utterance embeddings from Wiki-727k, created with SentenceBERT*[2] *(left) and HyperSeg (right). The color of each dot represents its topic. Representations were projected with t-SNE [4].*

This work exploits hyperdimensional computing (HDC) to create a robust topic segmentation system resilient to domain and hyperparameter change. HDC leverages vector interactions in high-dimensional spaces to create scalable, aggregated representations. We initialize each token in an utterance as a random $10,000$-dimensional vector and bind those token representations to portray a complete utterance. HDC's aptitude for single-pass learning makes it inherently robust to domain shift [3]. Compared to neural approaches, HDC produces more representative utterance embeddings in a fraction of the processing time (Figure 1). We summarize our contributions as follows:

1. We introduce HyperSeg, the first framework to use HDC for topic segmentation.
2. HyperSeg outperforms the best unsupervised transcript segmentation baselines across multiple domains, even when baselines are provided with optimal hyperparameters and partial ground truth.
3. HyperSeg is an order of magnitude faster than neural baselines, at an average of 10 times per utterance. HyperSeg achieves such speedup while running entirely on the CPU.
4. HyperSeg increases downstream summarization accuracy compared to baseline and naive segmentation schemes.

2. Related work

2.1. Hyperdimensional computing

The concentration of measure [7] is a phenomenon that occurs among a set of randomly initialized n-dimensional vectors where – given sufficiently large n – the cosine similarity between each vector is heavily concentrated around 0. Such a

[1] https://github.com/seongminp/hdseg

[2] https://huggingface.co/sentence-transformers/all-mpnet-base-v2. SentenceBERT [5] performed better than more recent SimSCE [6].

10.21437/Interspeech.2023-1859

set of vectors are said to be pseudo-orthogonal to each other. In other words, a random vector drawn in high-enough dimensions will be probabilistically dissimilar to those previously drawn. Hyperdimensional computing (HDC) [8] is a class of vector symbolic architectures that leverages such convenient orthogonality in high-dimensional space. HDC designates a random large dimensional vector (with $n >= 10,000$) to represent concepts of varying granularity, such as a single text token, a sentence, or an entity class.

HDC occupies the middle ground between symbolic artificial intelligence (e.g. agent-based modeling) and distributional artificial intelligence (e.g. neural networks), amending the shortcomings of both. In symbolic architectures, one-hot entity representations render entity similarity calculations discontinuous and thus non-interpolable [9, 10]. At the opposite end of the spectrum, distributional semantics uses vector representations for natural interpolation and loss propagation, but cannot consolidate learned vectors into order- and dimension-preserving sequences (known as the “binding problem” [11]). HDC retains the benefits of distributed vector representations while offering a solution to the binding problem by tapping into the aforementioned pseudo-orthogonality of vectors at high dimensions. At extremely high vector spaces, the pseudo-orthogonality of randomly realized vectors converges to exact orthogonality. The architecture also comes with the benefit of extremely fast and parallelizable representation construction.

HDC has seen adoption in time-sensitive real-time systems [9], often in conjunction with artificial neural networks. The framework has found its place in text processing as well [12], boasting high accuracy and extremely fast processing time. For the first time, we apply HDC’s representational power to transcript segmentation. We find that utterance embeddings created with HDC are semantically more coherent than ones created with the best sentence embedding neural networks.

2.2. Unsupervised topic segmentation

SOTA unsupervised topic segmentation algorithms first construct vector representations of each transcript utterance, to be delivered to downstream sequence segmentation algorithms. BERTSeg [13] and CohereSeg [14] use SentenceBERT [5] embeddings to represent each utterance. BERTSeg uses off-the-shelf pre-trained BERT [15] models, while CohereSeg further trains BERT with utterance-pair coherence loss. Semantic utterance vectors are fed to the TextTiling algorithm [16], which is a linear sequence segmentation scheme that determines topic segments by vector distance. GraphSeg [17] uses an alternative segmentation scheme by constructing a semantic graph with each utterance as a node. Utterance communities are determined in the semantic graph, and maximal cliques in the graph are designated as transcript segments.

Depending on the transcript domain, however, the representational capability of semantic embeddings can falter. Each utterance embedding contains an unexplainable juxtaposition of semantic and lexical concepts [1, 18]. We find existing transcript segmentation algorithms are brittle in both representation construction and downstream segmentation. Such brittleness makes SOTA baselines extremely sensitive to hyperparameters, often resulting in lower accuracy than random segmentation. We show that replacing the utterance embedding step with HDC produces more topic-aware utterance vectors across multiple datasets. Leveraging such representational advantages, HyperSeg eliminates all hyperparameters while outperforming previous SOTA, even when optimal hyperparameters (including partial ground truth, such as the total segment count of a transcript) are provided to baselines. HyperSeg also operates an order of magnitude faster than neural baselines, which are often unfit for resource-constrained preprocessing tasks such as topic segmentation.

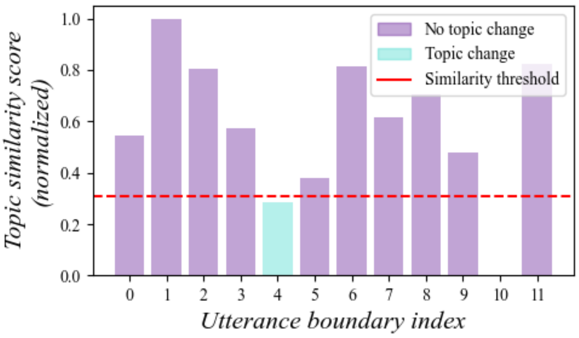

Figure 2: *Normalized similarity scores from Doc2Dial. Utterance boundaries with similarity under an adaptively determined threshold are considered topic boundaries.*

Table 1: *Summary of benchmark datasets. ”Utts” and ”Utt len” stand for mean utterance count and mean utterance length.*

Name	**Domain**	**Utts**	**Utt len**	**Segments**
AMI	Meeting	364.10	36.88	6.82
ICSI	Meeting	879.51	50.84	16.12
Doc2Dial	Customer service	30.13	77.57	1.42
VT-SSum	Lecture	306.10	82.72	33.87
Wiki-727k	General	46.43	125.36	4.66

3. Segmentation with HyperSeg

3.1. Constructing utterance embeddings

We follow [12] and create utterance embeddings by binding multiple hyperdimensional word embeddings. For every word in an transcript that is not a stopword, we randomly initialize a $10,000$-dimensional bipolar vector $\boldsymbol{w}$:

$$\begin{gathered}\boldsymbol{w} = (x_0, \ldots, x_{9,999}), \\ \text{where } x \sim \text{IID}\{-1, 1\}.\end{gathered} \tag{1}$$

Each element in the word embedding is either -1 or 1, and same words are mapped to the same embedding. High dimensionality ensures that each word embedding is roughly orthogonal to every other word embedding (“blessing of dimensionality” [19]).

To represent a single utterance, we use permutation binding [9, 12] to consolidate all word embeddings in the utterance into a single hyperdimensional vector, also of dimension $10,000$. To generate an utterance vector $\boldsymbol{u}$, we first encode the position of each constituent word vector $\boldsymbol{w_i}$ by right-shifting (Π) $l - i - 1$ times. l is the utterance word count and i is the zero-based word index within $\boldsymbol{u}$. The final utterance embedding is the component-wise majority vote (M) of all of its position-

Table 2: *Transcript segmentation results. HyperSeg performs best even without the ground-truth segment count provided to baselines. "BS" stands for boundary similarity. Results are averaged over 5 runs with different random seeds. Higher is better for both metrics.*

	AMI		**ICSI**		**Doc2Dial**		**VT-SSum**		**Wiki-727k**	
Model	*F1*	*BS*	*F1*	*BS*	*F1*	*BS*	*F1*	*BS*	*F1*	*BS*
Random	5.19	2.75	5.09	2.64	10.76	6.89	23.91	13.30	28.37	16.60
GraphSeg [17]	4.22	2.22	2.75	1.43	10.06	5.35	2.28	1.18	21.09	12.26
BERTSeg [13]	4.43	2.36	1.58	0.81	2.95	1.56	8.84	4.75	17.91	10.23
CohereSeg [14]	8.77	4.63	**7.79**	**4.03**	16.58	10.09	25.98	14.57	28.41	**17.60**
HyperSeg (ours)	**9.41**	**4.93**	6.94	3.55	**18.49**	**10.54**	**27.35**	**15.15**	**29.82**	17.45

encoded word vectors ($\boldsymbol{w_i^{pos}}$):

$$\boldsymbol{w_i^{pos}} = \Pi^{l-i-1}(\boldsymbol{w_i}) \tag{2}$$

$$\boldsymbol{u} = M(\boldsymbol{w_0^{pos}}, \boldsymbol{w_1^{pos}}, ...\boldsymbol{w_{l-1}^{pos}}) \tag{3}$$

Topic segmentation does not require extracting constituent word embeddings after obtaining utterance embeddings. Even so, the reversibility of right shifts and position-wise multiplication guarantees the ability to recover each word embedding should the need arise.

3.2. Transcript segmentation

We calculate the boundary similarity score (S) for every neighboring utterance pair $\boldsymbol{u_i}$ and $\boldsymbol{u_{i+1}}$:

$$S = \frac{\boldsymbol{u_i} \cdot \boldsymbol{u_{i+1}}}{|\boldsymbol{u_i}||\boldsymbol{u_{i+1}}|} \tag{4}$$

Each boundary score is the cosine similarity between surrounding utterance embeddings. An utterance boundary index j has to satisfy two conditions to be selected as a topic boundary: its similarity score s_j has to be a local minimum, and s_j has to be lower than an automatically determined threshold T.

Since the desired number of total segments is unknown, selecting a similarity threshold for topic boundaries is a non-trivial process. Baselines rely on hyperparameters to determine the total segment count, which often hinders segmentation accuracy at test time. HyperSeg automatically calculates the similarity cutoff threshold T as one standard deviation of boundary scores (σ_s) subtracted from the mean (μ_s). We design T in such a way to incorporate both local (cosine distance between neighboring embeddings) and global (mean and standard deviation of all distances) statistics. The number of topics in a document is therefore adaptively determined.

Local minima boundary indices from S's index set $I(S)$ with scores below T are selected as the set of final segment boundaries B (Figure 2):

$$T = \mu_s - \sigma_s \tag{5}$$

$$B = \{j \in I(S) | s_j \text{ is a local mimimum} \wedge s_j < T\} \tag{6}$$

HyperSeg can be optionally operated in *damp mode*, where the number of segments is capped with a logarithm filter. Of utterance boundaries with similarity smaller than T, we select N boundaries with the smallest boundary cosine similarity, where

$$N = \lfloor \log^2 |\{i \mid scores_i < T\}| \rfloor \tag{7}$$

Dampened HyperSeg creates segments less eagerly.

4. Experiments

We measure HyperSeg's effectiveness with two approaches: raw segmentation performance (accuracy and speed) and improvements in downstream summarization of segmented text. We also compare simple word- and n-gram counting with HyperSeg to empirically illustrate how HyperSeg performs beyond naive surface-level lexical matching. We use GraphSeg [17], BertSeg [13], and CohereSeg [14] as state-of-the-art baselines.

4.1. Datasets and metrics

Segmentation accuracy of HyperSeg and baselines are tested on five benchmarks: **AMI** [20], **ICSI** [21], **Doc2Dial** [22], **VT-SSum** [23], and **Wiki-727k** [24]. All datasets except Wiki-727k consist of dialogue transcripts. We include Wiki-272k to demonstrate that HyperSeg's robustness extends to written text segmentation. Selected benchmarks cover diverse domains, including meetings, customer service, lectures, and others. Table 1 includes detailed statistics of each dataset.

Linear segmentation is evaluated as a binary classification problem for each possible utterance boundary. Segmentation accuracy is measured with classification F1 scores and Boundary Similarity [25]. The F1 measure captures the harmonic mean between precision and recall for the binary classification problem (boundary-or-not). Boundary Similarity measures the inter-coder agreement between system and reference segmentations.

Downstream summarization accuracy is measured by how much the system summary overlaps with the ground-truth summary. We measure ROUGE-1 (**R1**), ROUGE-2 (**R2**), and ROUGE-L (**RL**) [26] between reference and system summaries. An off-the-shelf BART [27] model trained on the standard CNN/Daily Mail dataset [28] is used as the downstream summarizer.

5. Results

5.1. Transcript topic segmentation

HyperSeg outperforms baselines in all datasets except ICSI, even when the compared systems use the most optimal hyperparameters (Table 2). For GraphSeg, BertSeg, and CohereSeg, we report their best possible scores by providing the true number of ground-truth segments as a hyperparameter. Only HyperSeg and CohereSeg consistently outperform random segmentation. Performance of all systems noticeably degrades in AMI and ICSI, where true segment boundaries constitute less than 2% of all possible breakpoints. For such sparse datasets where metrics fail to take near-misses into account, we also report downstream summarization performance in the next section. All reported results are averages after 5 runs with different random seeds.

Table 3: *Average milliseconds elapsed per utterance. HyperSeg is 10 times faster than the next fastest baseline.*

Model	AMI	ICSI	Doc2Dial	VT-SSum	Wiki-727k	Mean	× HyperSeg
GraphSeg [17]	11.60	13.71	10.66	11.11	9.73	11.36	× 10.42
BERTSeg [13]	11.92	12.02	12.82	11.45	11.22	11.89	× 10.91
CohereSeg [14]	15.99	14.75	15.15	15.28	15.76	15.39	× 14.12
HyperSeg (ours)	0.77	0.70	1.19	1.28	1.49	1.09	× 1

On average, HyperSeg is 10 times faster per utterance than the next quickest baseline GraphSeg (Table 3). Reported segmentation speeds are measured on an AMD EPYC 7742 CPU and four 80GB A100 GPUs. All baseline runs were conducted on GPUs, while HyperSeg ran entirely on the CPU.

5.2. Downstream summarization performance

To demonstrate the effect of accurate topic segmentation on downstream summarization, we apply HyperSeg to AMI and ISCI before summarizing the transcripts. Compared to no segmentation, naive segmentation of equal segment length, and random segmentation, HyperSeg always improves summarization in terms of ROUGE scores (Table 4).

Table 4: *Downstream summarization accuracy. Only HyperSeg yields consistently better ROUGE scores than random or naive baselines.*

	AMI			ICSI		
Model	*R1*	*R2*	*RL*	*R1*	*R2*	*RL*
None	0.19	0.03	0.12	0.17	0.03	0.13
Random	0.27	0.05	0.13	0.16	0.02	0.08
Uniform	0.13	0.01	0.08	0.22	0.03	0.12
GraphSeg [17]	0.22	0.04	0.13	0.23	0.04	0.12
BERTSeg [13]	0.22	0.04	**0.14**	0.21	0.04	0.11
CohereSeg [14]	0.27	**0.06**	0.12	0.15	0.04	0.08
HyperSeg	**0.32**	**0.06**	**0.14**	**0.30**	**0.08**	**0.15**

5.3. Difference to simple lexical matching

HyperSeg offers two pronounced advantages over simple word or n-gram count-based topic segmentation: fine-grained sentence similarity scores and flexibility in token granularity. The distributed nature of HyperSeg produces a continuous spread of utterance boundary similarities (Figure 3). Word or n-gram counting yields a sparse spread of utterance similarities. Such segmentation renders the vectors unusable as features in possible downstream classification networks. HyperSeg can also accommodate character-level tokenization by modifying its binding algorithm. Such change in granularity offers a trade-off between robustness against character error rate and representational burden (an utterance vector has to contain more token vectors within a designated dimension compared to word-level tokenization).

6. Conclusion

HyperSeg is a new state-of-the-art topic segmentation algorithm. We find that hyperdimensional computing's representational benefits can be effectively realized in dialogue topic segmentation. HyperSeg is more accurate and lightweight compared to best-performing baselines. We attribute HyperSeg's performance to the robustness of HDC embeddings across domains. Such advantages make HyperSeg an ideal component in ASR post-processing, which is often resource-constrained. We confirm that HyperSeg's representational capabilities extend to downstream summarization as well.

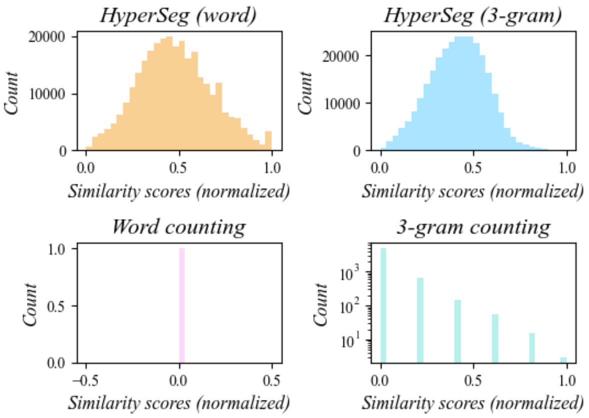

Figure 3: *Distribution of utterance vector similarities (VT-SSum). Regardless of tokenization scheme (word-level or 3-gram), HyperSeg produces more gradual utterance similarity vectors compared to word or n-gram counting that use one-hot token vectors.*

While this paper deals solely with linear transcript segmentation, a more natural approach to dialogue segmentation would take repetitive alternations of topics into consideration. Such methods require gathering related utterances into non-linear topical clusters. Equipped with the representational robustness of hyperdimensional vectors, HyperSeg can be naturally extended to accommodate such advanced schemes for segmentation. We also leave experiments with different utterance tokenization schemes (character-level, subword-level) for future research.

7. References

[1] Z. Wang, D. Mekala, and J. Shang, "X-class: Text classification with extremely weak supervision," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 3043–3053.

[2] X. Wang, H. Wang, and D. Yang, "Measure and improve robustness in NLP models: A survey," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 4569–4586. [Online]. Available: https://aclanthology.org/2022.naacl-main.339

[3] J. Morris, K. Ergun, B. Khaleghi, M. Imani, B. Aksanli, and T. Simunic, "Hydrea: Utilizing hyperdimensional computing for a more robust and efficient machine learning system," *ACM Trans. Embed. Comput. Syst.*, vol. 21, no. 6, oct 2022. [Online]. Available: https://doi.org/10.1145/3524067

[4] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[5] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. [Online]. Available: https://aclanthology.org/D19-1410

[6] T. Gao, X. Yao, and D. Chen, "SimCSE: Simple contrastive learning of sentence embeddings," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6894–6910. [Online]. Available: https://aclanthology.org/2021.emnlp-main.552

[7] M. Ledoux, *The concentration of measure phenomenon*. American Mathematical Soc., 2001, no. 89.

[8] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive computation*, vol. 1, pp. 139–159, 2009.

[9] D. Kleyko, D. Rachkovskij, E. Osipov, and A. Rahimi, "A survey on hyperdimensional computing aka vector symbolic architectures, part ii: Applications, cognitive models, and challenges," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–52, 2023.

[10] G. Marcus, "The next decade in ai: four steps towards robust artificial intelligence," *arXiv preprint arXiv:2002.06177*, 2020.

[11] K. Greff, S. Van Steenkiste, and J. Schmidhuber, "On the binding problem in artificial neural networks," *arXiv preprint arXiv:2012.05208*, 2020.

[12] F. R. Najafabadi, A. Rahimi, P. Kanerva, and J. M. Rabaey, "Hyperdimensional computing for text classification," in *Design, automation test in Europe conference exhibition (DATE), University Booth*, 2016, pp. 1–1.

[13] A. Solbiati, K. Heffernan, G. Damaskinos, S. Poddar, S. Modi, and J. Cali, "Unsupervised topic segmentation of meetings with bert embeddings," *arXiv preprint arXiv:2106.12978*, 2021.

[14] L. Xing and G. Carenini, "Improving unsupervised dialogue topic segmentation with utterance-pair coherence scoring," in *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Singapore and Online: Association for Computational Linguistics, Jul. 2021, pp. 167–177. [Online]. Available: https://aclanthology.org/2021.sigdial-1.18

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[16] M. A. Hearst, "Text tiling: Segmenting text into multi-paragraph subtopic passages," *Computational linguistics*, vol. 23, no. 1, pp. 33–64, 1997.

[17] G. Glavaš, F. Nanni, and S. P. Ponzetto, "Unsupervised text segmentation using semantic relatedness graphs," in *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 125–130. [Online]. Available: https://aclanthology.org/S16-2016

[18] T. Pham, T. Bui, L. Mai, and A. Nguyen, "Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks?" in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 1145–1160.

[19] A. N. Gorban and I. Y. Tyukin, "Blessing of dimensionality: mathematical foundations of the statistical physics of data," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 376, no. 2118, p. 20170237, 2018.

[20] W. Kraaij, T. Hain, M. Lincoln, and W. Post, "The ami meeting corpus," 2005.

[21] A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke *et al.*, "The icsi meeting corpus," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, vol. 1. IEEE, 2003, pp. I–I.

[22] S. Feng, K. Fadnis, Q. V. Liao, and L. A. Lastras, "Doc2dial: a framework for dialogue composition grounded in documents," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 09, 2020, pp. 13 604–13 605.

[23] T. Lv, L. Cui, M. Vasilijevic, and F. Wei, "Vt-ssum: A benchmark dataset for video transcript segmentation and summarization," *arXiv preprint arXiv:2106.05606*, 2021.

[24] O. Koshorek, A. Cohen, N. Mor, M. Rotman, and J. Berant, "Text segmentation as a supervised learning task," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 469–473. [Online]. Available: https://aclanthology.org/N18-2075

[25] C. Fournier, "Evaluating text segmentation using boundary edit distance," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Sofia, Bulgaria: Association for Computational Linguistics, Aug. 2013, pp. 1702–1712. [Online]. Available: https://aclanthology.org/P13-1167

[26] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81. [Online]. Available: https://aclanthology.org/W04-1013

[27] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 7871–7880. [Online]. Available: https://aclanthology.org/2020.acl-main.703

[28] R. Nallapati, B. Zhou, C. dos Santos, Ç. Gulçehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence rnns and beyond," in *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, 2016, pp. 280–290.