https://doi.org/10.2352/EI.2024.36.10.IPAS-252

Automatic Reading Order Sequencing: A Novel Reading Order Generator for Text-Based Documents

Md. Manzoor Murshed, Steven J. Simske, Arturo N. Villanueva, Jr.
Colorado State University, Fort Collins, Colorado, United States

Abstract

There are many electronic documents salient to read for each given topic; however, finding a suitable reading order for pedagogical purposes has been underserved historically by the text analytics community. In this research, we propose an automatic reading order generation technique that can suggest a suitable and optimal reading order for curriculum generation quantitatively. It is necessary to read the relevant documents in some logical order to understand the topics clearly. There are many learning pedagogies advanced, so for our purposes we use the author-supplied reading orders of salient content sets for ground truth. Our method suggests the best reading order automatically by checking the relevant topics, document distances, and semantic structure of the given documents. The system will generate a suitable and efficient reading sequence by analyzing the information, similarity, overlap of contents, and distances using word frequency, and topic sets. We measure the similarity, relevance, distance, and overlap of different documents using cosine similarity, entropy relevance, Euclidean distances, and Jaccard similarities respectively. We propose an algorithm that will generate the best possible reading order for a set of given documents. We evaluated the performance of our system against the ground truth reading order using different kinds of textbooks and generalized the finding for any given set of documents.

Introduction

The purpose of our research is to generate the best possible reading order for a given set of documents on related topics. Reading a set of documents or even a given textbook has the purpose of acquiring knowledge and gaining information. Readers can perform full, superficial, or even shallow reading of the content. With full reading, they can gain most of the knowledge and information, however, superficial and shallow reading will provide some of the knowledge and information from the text. As the reading will increase cognition, a full reading of all the materials will provide the highest information to the reader. We are trying to find the best possible reading order so that users can gain all the knowledge from the given documents. Our research also can suggest the best possible reading order for a set of search engine results. Textbook authors usually present materials in some logical order. It is not always necessary to read the contents in the same logical order. For the sake of curriculum, authors also suggest different reading orders due to the prerequisite and convenience. However, to measure the suitability of our sequence, we used the author's generated sequence of a textbook or course materials as a ground truth golden standard.

In this research, we proposed a method to automatically generate a suitable reading sequence for curriculum generation using different kinds of relationships of the documents. Generating such a sequence automatically will be very helpful for educators, publishers, researchers, or even general readers as our method is based on quantitative relationships of the documents. Readers can read the documents in some logical sequence, which might save unnecessary searching and ordering of the documents.

We propose automatically determining the reading sequence based on four types of relationships between documents. These four types of relationships are similarity, overlap, distances, and generality. We can use different kinds of methods for calculating those relationships, however, for presenting our methods we used Cosine similarity, Jaccard distance, Euclidean distance, and Shannon's Entropy respectively. Entropy, an important topic in probability and statistics has been used as it has strong connections with information theory [5]. A set of documents are equivalent when they present same topics at the same level or slightly different level. On the other hand, a set of documents are more specific or general when they present different topics at different level. We tried to capture these relationships mathematically using those metrics. When a set of documents are equivalent, we can sequence them in any order, but a more specific or general document will be better to read sequentially. We have used Topic Modeling, the most common application of natural language processing for document clustering and organizing large collections of data in text format [9]. For example, the introductory chapter of a textbook or dissertation is generally more specific or general as it tries to present a summary of the topics that will be covered in the subsequent chapters. Similarly, later chapters could be equivalent, or they might have some prerequisite relationships with each other. We tried to capture the topical similarity, using Cosine distance, the overlap of the topical content using Jaccard distance, the closeness of the topics using Euclidean distance, the and generality of the topics using Entropy. Our main goal is to propose an automated technique to logically sequence a set of documents using only the words that appear in the corpus. As we assume that the set of documents is logically related to each other, therefore the metrics to calculate distances will reflect the closeness of the presented topics. We will try to find the optimal reading order using those distance relationships only. We evaluate the performance of our sequencing methods using a set of data of known logical sequence. The methods that we describe in this research are applicable to different documents written on the same topic, course materials that are logically connected, or even to finding the most relevant documents that match a user's query and ranking those documents by relevance.

Automatic reading order generation might be beneficial in many areas. Educational institutes or educators can use this to automatically generate a suitable curriculum, researchers and general readers can use this to find a suitable reading order based on their topic needs, and publishers can organize the research materials in a more logical order. We examined our methods for a set of documents with known logical sequences like textbooks, course materials, and dissertations, where chapters are somewhat logically sequenced by the authors. Our sequencing methods may potentially be useful to sequence curriculum documents, sequence research documents on a particular topic, logically sequence journal articles, find the logical sequences of a related news story, etc, among others. Such applications can be helpful in many fields such as Linguistics, Document retrieval, Data mining and clustering, and for generating better search results for a search engine. The contributions of our work are the following.

1. We presented a suitable algorithm to automatically sequence a set of documents in a logical order.

2. We proposed different metrics to capture the underlying relationships of documents suitable for sequencing.

3. We evaluate the performance of the algorithm using some suitable metrics.

4. We tested our method with textbooks, and course materials and tested the validity of our concept.

Relevant Research

Generating Reading Orders Over Document Collections [4]: This paper presented a technique to generate a reading order for a set of documents using a tree structure where the nodes contain a set of equivalent documents and edges represent a reading order from general to more specific documents. They quantitatively define document generality and document overlap concept. By using these metrics, they presented an algorithm to read general documents to more focused documents by using only the document contents that require no external knowledge about the documents. The generated reading tree capture document specificity relationships and can generate multiple reading orders of the documents

On Statistical Sequencing of Document Collections [8]: In this paper Thinniyam proposed two techniques for distance-based methods for sequencing. He found that Latent Semantic Analysis (LSA) based distance between documents is more effective than the Tableaux-based distancing method that uses similarity and distance measurements to find sequence elements that are similar

A data mining approach to reading order detection [2]: This paper investigates the reading order problem by resorting to data mining techniques that can generate the required knowledge from a set of training layout structures by using a naive Bayesian classification framework. This paper also provides solutions for reconstructing the chain over a block of a layout.

Other authors presented a reading order detection approach "LayoutReader" in a single document using deep neural networks [10]. They also published a benchmark 500,000 document image dataset "ReadingBank" for reading order detection research. While their work is to find the reading order of a single document, we presented a technique to find the reading order sequence of a collection of documents.

Pre-processing

We represent the set of n documents to be sequenced D = {D1, D2, … , Dn} with m unique words or terms T = {t1, t2, . . . , tt} that appear in the corpus. The purpose of preprocessing is to convert n documents to t terms, t1 … tt , to generate matrix Dn x t which consists of the term frequencies of the given documents. The pre-processing step will remove stop words and perform stemming and lemmatization of the documents. We represent this document term matrix D in two formats, one with purely word frequencies and the other with tfidf, which is term frequency (tf) times inverse document frequency (idf) of the given set of n documents. The tfidf representation is generally used in document classification and we wanted to see the difference and quality of our generated sequence using these two methods of representation of corpus. For calculating the idf we used the formula idf(t) = log [(1 + n) / (1 + df(t))] + 1, where n is the number of documents in the corpus, and document frequency (df) is the number of documents in the document set that contain the term t. The effect of adding 1 is not to ignore terms that occur in every document, and the constant 1 is added to the numerator and denominator to prevent zero divisions (scikit-learn.org). The tfidf representation of the documents set can scale down the impact of tokens that occur very frequently in a given corpus because they are empirically less informative than features that occur in a small fraction of the training corpus. The output of this step is a document-term matrix that maps all documents to a fixed term frequency of words that comprise the corpus.

For clustering, the concept of K-means has been used. In the k-means technique, the prototype is defined in terms of the mean of a group of points (known as a centroid) and is applied to objects in a continuous multi-dimensional space [7]. From the document terms matrix D, using the k-mean clustering algorithm we cluster all the t terms in k clusters. We used these k clusters as k topics of interest, where each topic consists of a collection of words from the corpus. The value of K is always less than or equal to the number of documents n. Using the Document term matrix D and Topics table T, we generate the Document topics matrix M, which comprises the percentage of the topics that are discussed in every given document in D. Document topic matrix explains how k topics are distributed in the n documents. From our D (Table 1) and T (Table 2) matrices, we generate the following document topic matrix M. Document topic matrix Mnxk represents the distribution of k topics in n documents where Ti is the topic vector for document Di, which represents the distribution of all the topics in document Di. We can assume that Mim ε [0, 1] is the probability that topic km is associated with document Di, where i ≤ n and m ≤ k.

Each document in document set D represents some information. We calculate the overall information score of each of the documents using the Shannon entropy equation, where higher document entropy is the indication that more topics are covered in the document. For a given document Di, we calculate the document information score I(Di) using Equation 1:

$$I(D_i) = H(D_i) = \sum - M_{im} \log(M_{im}) \qquad (1)$$

As all the documents are represented as fixed topics of given terms, there are some similarities and overlaps of terms between them. We calculated the similarity of the documents using cosine similarity and document overlaps using Jaccard distance. The cosine similarity of the two documents is calculated using the equation: Cosine similarity C (A, B) = A.B /(|A||B|).

Therefore, document similarity matrix C: Dn x Dn mapped to [0, 1], represents the similarity between two documents. For curriculum generation and reading order generation we will use the concept of similarity.

We calculate the overlap score of a pair of documents using the Jaccard distance, where,

$$O(D_i, D_j) = Jaccard(D_i, D_j) = \frac{D_i . D_j}{|D_i|^2 + |D_j|^2 - D_i . D_j}$$

We calculate the pairwise distance between two documents using the Euclidean distance.

E(X, Y) = Euclidean (X, Y) = $(\sqrt[2]{\sum_{i=1}^{n}(x_i - y_i)^2}$ .Euclidean distance represents the relative distances between the documents.

Evaluation methods

After generating the ordering sequence, we estimate the accuracy of the estimated ordering using different methods. If we have n documents to be ordered, we can order them in n! ways.

Euclidean distance

If we have two documents order X= [$x_1, x_2, \ldots, x_n$], where x_i is the ith position of a document, then the Euclidean distance between X and Y can be expressed with the equation Euclidean (X, Y) = $(\sqrt[2]{\sum_{i=1}^{n}(x_i - y_i)^2}$. If the two sequences are same order, then Euclidean distance is 0.

Manhattan distance

The Manhattan distance is defined as the sum of the absolute values of the differences between two orders. For two document order X and Y Manhattan distance $MD(X, Y) = \sum_{i=1}^{n}|x_i - y_i|$, for the above value of X and Y, MD(X, Y) = 4. If the two sequence are same order, then Manhattan distance is 0.

Spearman Rank-Order Correlation (ρ)

This metric can explain the better relationship and order between two variables (X and Y), which is the Pearson correlation coefficient between the ranked variables. If we represent our n ordered sequence of documents with X, where xi is the rank, where highest value is ranked 1 and the lowest value is ranked n. Then the correlation can be computed from these ranked values using the formula:

$$\rho(X, Y) = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 (y_i - \bar{y})^2}}}$$

This equation can represent a perfect correlation as +1 and complete reversal of the order as -1, where positive value indicates that the ordering follows the same direction and negative value indicates the opposite. Suppose we have 5 documents in the following three orders X = [1, 2, 3, 4, 5], Y = [2, 3, 1, 4, 5], and Z = [5, 4, 3, 2, 1]. Then, if X is our true sequence, Euclidean (X, Y) = 2.45, Euclidean (X, Z) = 6.32, MD(X, Y) = 4, MD(X, Z) = 12, ρ(X, Y)= 0.7, and ρ(X, Z)= -0.99. All the three metrices indicate that the sequence Y is better and closer to the real sequence. However, while Euclidean distance, and Manhattan distance gave high value for the reverse sequence, the negative value of the Spearman Rank-Order can correctly represent that the order Z is not a better sequence.

System Architecture

Our proposed system can be represented with the block diagram in Figure 1. The set of n documents are preprocessed by removing the stop words, lemmatization, and stemming. The preprocessed documents are represented by the bag-of-words using frequencies and tfidf values. The system used those vectors to calculate different distances and topic modeling. The sequence algorithm used those distances and topics to generate the output sequence of the documents. Finally, the system evaluates the generated sequence using different metrics.

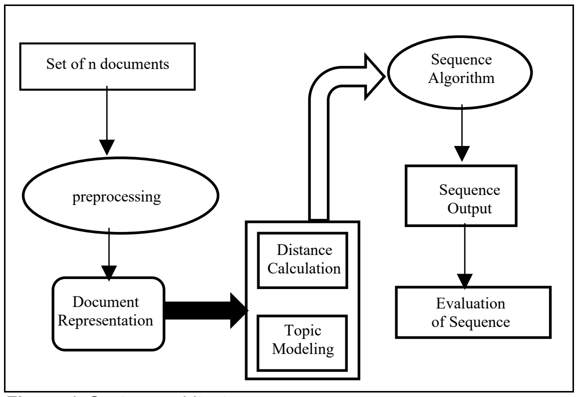

Figure 1. *System architecture*

Reading order algorithm

Our algorithm is presented in the following pseudocode.

Table 1: Automatic Reading Order Generation Algorithm

```
Algorithm: Reading order generation
Input: document set D
Output: Full reading order of the document
    Step 1: for each Di in D
            calculate document Entropy, topic matrix T,
            and information score I
            document overlap matrix M,
            document similarity matrix C,
            and document distance scores E
        end for
    Step 2: Pickup the document with the highest document entropy.
    Step 3: while while(D<>0) do
if next_doc has highest jaccard & cosine, and lowest euclidean
        next sequence = next_doc
else if next_doc has highest jaccard and cosine
        next sequence = next_doc
else if next_doc has highest jaccard and lowest euclidean
        next sequence = next_doc
```

```
else if next_doc has highest cosine and lowest euclidean
        next sequence = next_doc
else
        next sequence = next_doc with the highest entropy
                difference
  end while
    Step 4: End
```

The rationale for picking up the document with the highest document entropy is that, we assume that such a document contains most of the information and summary of the topics. The high overlap, high similarity, and low distance document or document with any two of these three criteria will choose the most similar topic document that is logically connected. In case, if we failed such a document, we choose the following document that has the highest information scores, because it might represent some new topics and information in the next sequence.

Example

Using the above algorithm let us try to generate the automatic reading sequence of the textbook "Functional Applications of Text Analytics Systems" by Steven Simske [6]. The textbook consists of seven chapters presented in table 2.

Table 2: Sample document sets from the textbook

Set of Documents	Chapter title	Number of words
Chapter 1	Linguistics and NLP	13523
Chapter 2	Summarization	12960
Chapter 3	Clustering, Classification, and Categorization	12960
Chapter 4	Translation	13491
Chapter 5	Optimization	12741
Chapter 6	Learning	13943
Chapter 7	Testing and Configuration	8914

After preprocessing the documents, we generate the document term matrix where each chapter is represented by a vector of word frequencies and a vector of tfidf values.

Table 3: Document topic matrix T

	T1	T2	T3	T4	T5	T6	T7	Ent.
c1	7.20	8.03	8.24	6.24	7.61	7.00	14.58	0.827
c2	7.18	5.70	7.20	5.80	13.29	7.06	6.68	0.825
c3	6.67	6.30	8.09	14.3	5.87	5.88	5.79	0.819
c4	4.88	7.81	5.74	5.11	5.83	13.05	6.42	0.818
c5	7.82	8.09	15.45	7.12	7.18	7.47	9.18	0.827
c6	13.67	5.63	6.15	6.33	6.95	6.70	6.51	0.822
c7	6.89	14.88	8.65	6.41	5.90	8.32	7.86	0.823

We apply the k-mean clustering algorithm to generate the document term matrix of Table 3 and we calculate the entropy of each chapter from this table. Using the Document topic matrix we generate the Jaccard similarity table (Table 3), Cosine similarity table (Table 4), and Euclidean distance table (Table 5) below.

Table 4: Document overlap matrix M

	ch01	ch02	ch03	ch04	ch05	ch06	ch07
ch01	1	0.485	0.426	0.478	0.48	0.461	0.415
ch02	0.485	1	0.47	0.483	0.468	0.469	0.401
ch03	0.426	0.47	1	0.411	0.447	0.428	0.405
ch04	0.478	0.483	0.411	1	0.456	0.461	0.409
ch05	0.48	0.468	0.447	0.456	1	0.466	0.429
ch06	0.461	0.469	0.428	0.461	0.466	1	0.401
ch07	0.415	0.401	0.405	0.409	0.429	0.401	1

Table 5: Document similarity matrix C

	ch01	ch02	ch03	ch04	ch05	ch06	ch07
ch01	1	0.463	0.334	0.432	0.545	0.418	0.519
ch02	0.463	1	0.285	0.314	0.433	0.364	0.325
ch03	0.334	0.285	1	0.246	0.462	0.331	0.346
ch04	0.432	0.314	0.246	1	0.395	0.309	0.469
ch05	0.545	0.433	0.462	0.395	1	0.449	0.564
ch06	0.418	0.364	0.331	0.309	0.449	1	0.358
ch07	0.519	0.325	0.346	0.469	0.564	0.358	1

Table 6: Document distance scores E

	ch01	ch02	ch03	ch04	ch05	ch06	ch07
ch01	0	1.036	1.154	1.066	0.954	1.079	0.98
ch02	1.036	0	1.196	1.171	1.065	1.128	1.162
ch03	1.154	1.196	0	1.228	1.038	1.157	1.144
ch04	1.066	1.171	1.228	0	1.1	1.175	1.031
ch05	0.954	1.065	1.038	1.1	0	1.05	0.933
ch06	1.079	1.128	1.157	1.175	1.05	0	1.133
ch07	0.98	1.162	1.144	1.031	0.933	1.133	0

The algorithm of Table 1 generated the following reading sequence as output (Table 7). If we assume the author-generated sequence as a ground truth, the matrix in the table evaluated the generated sequences.

Table 7: Output of the system

Method	Output sequence	Euclidean distance	Manhattan distance	Spearman Correlation
With frequency	Ch1, Ch4, Ch7, Ch5, Ch3, Ch6, Ch2	7.071	14	0.107
With tfidf	Ch1, Ch5, Ch7, Ch4, Ch2, Ch6, Ch3	7.071	14	0.107

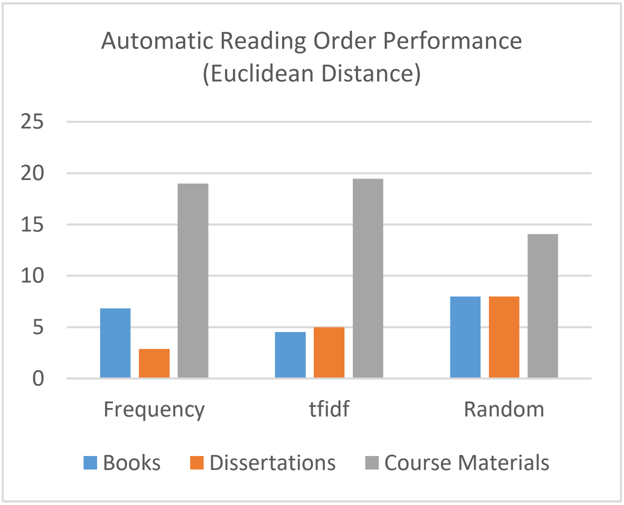

Figure 2. *Reading order performance (Euclidean Distance)*

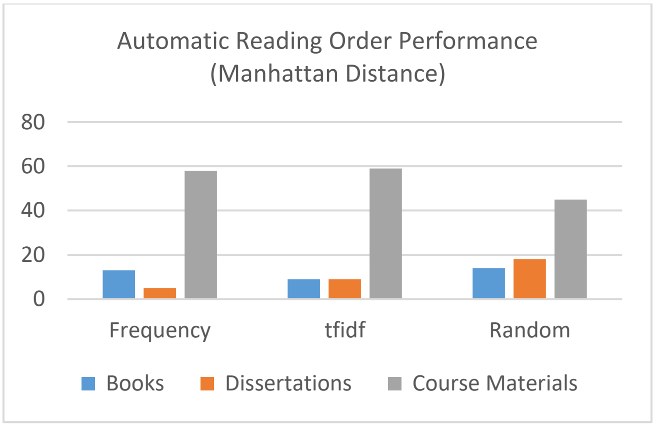

Figure 3. *Reading order performance (Manhattan Distance)*

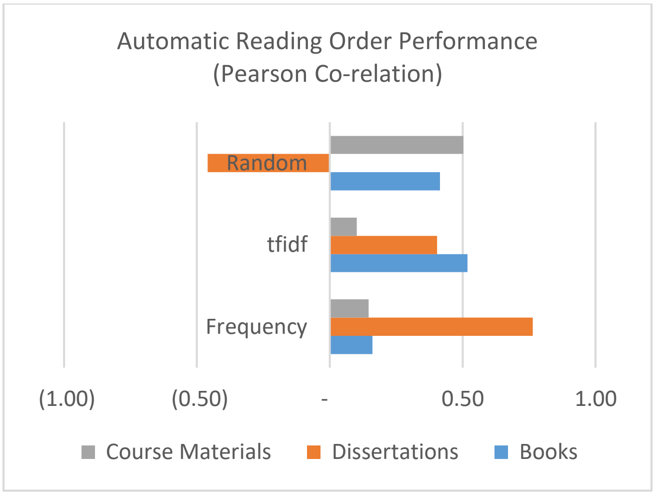

Figure 4. *Reading order performance (Pearson Co-relation)*

Results

We tested our algorithm on books, dissertations, and university courses. The average Euclidean distance, Manhattan distance, and Spearman correlation are presented in Table 7. We compare our results with randomly generated sequences. From the result, we can see that the system can generate meaningful sequences close to the real author-provided sequence and for all cases our algorithm is performing better than random sequences. The *tfidf* performs better for books corpora, whereas frequency representation performs better for the other two kinds of corpora. For Pearson Co-relation, we found that average random sequences for dissertations are negatively correlated. We can also observe that finding good sequences in courses is difficult, maybe due to the reason for using PowerPoint slides as corpora.

The result of our algorithm is presented in the graph of Figures 2, 3, and 4.

Conclusion

In this research, we presented an algorithm for generating reading order sequences automatically for a collection of related documents using the word features. Our work can predict document relationships like similarities, overlaps, and distances without using any external knowledge. This algorithm is capable of sequencing a collection of logically related text documents. We used different metrics for judging the result of our algorithm. We validated our finding by measuring the quality of the sequencing of books, dissertations, and course materials, where the original sequencing comes from the authors. As the proposed reading order is somewhat similar to the original sequencing, it can be concluded that the automated reading order sequencing algorithm is appropriate for any related text-based set of documents.

In future work, we will include the images and other visual information of the document features as well as other types of document relationships. For topic modeling in addition to k-means, we will try to use Latent Dirichlet Allocation [1] or other types of clustering. We will also try to propose methods to personalize reading sequences for different levels and different users with background knowledge.

References

[1] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. Journal of Machine Learning Research, 3, 993–1022. https://doi.org/10.5555/944919.944937

[2] Ceci, M., Berardi, M., Porcelli, G., & Malerba, D. (2007). A data mining approach to reading order detection. In *Proceedings of the . . . International Conference on Document Analysis and Recognition*. Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/icdar.2007.4377050

[3] Changuel, S., Labroche, N., & Bouchon-Meunier, B. (2015). Resources sequencing using Automatic Prerequisite--Outcome Annotation. *ACM Transactions on Intelligent Systems and Technology*, *6*(1), 1–30. https://doi.org/10.1145/2505349

[4] Koutrika, G., Liu, L., & Simske, S. J. (2015). *Generating reading orders over document collections*. https://doi.org/10.1109/icde.2015.7113310

[5] Serrano, L. (2021). Grokking Machine learning. Simon and Schuster.

[6] Simske, S., & Vans, M. (2022). Functional applications of text analytics systems. CRC Press.

[7] Tan, P., Steinbach, M., & Kumar, V. (2006). Introduction to data mining. Addison-Wesley.

[8] Thinniyam, R. On Statistical Sequencing of Document Collections. Ph.D. thesis, University of Toronto, 2014. 670

[9] Vajjala, S., Majumder, B., Surana, H., & Gupta, A. (2020). Practical natural language processing: A Pragmatic Approach to Processing and Analyzing Language Data. O'Reilly Media.

[10] Wang, Z., Xu, Y., Cui, L., Shang, J., & Wei, F. (2021). LayoutReader: Pre-training of text and layout for reading order detection. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. https://doi.org/10.18653/v1/2021.emnlp-main.389.

Author Biography

Md. Manzoor Murshed is a Ph.D. candidate in Systems Engineering at Colorado State University. Currently, he is a full-time faculty of the Computer Science Department at the North East Iowa Community College (NICC) and an adjunct professor of the Information Technology Department at Upper Iowa University (UIU). Prior to joining NICC, he served UIU as an Associate Professor for the last 22 years and used to teach Information Systems, Information Technology, and Software Engineering majors. He received a B.Sc (Honors) in Applied Physics and Electronics with Math and Chemistry as a Minor and M.Sc (Thesis) in Computer Science from Dhaka University. He also earned an M.S. in Computer Engineering and a Certificate in Information Assurance from Iowa State University and an MBA from Upper Iowa University.

Dr. Steve Simske is a Professor in Systems Engineering, and an affiliate of Biomedical and Mechanical Engineering, at Colorado State University (CSU). The author of 230 US patents and more than 450 publications, he is an IS&T, IEEE, and NAI Fellow. Steve is the Steering Committee Chair for the ACM DocEng Symposium, and a past member of the World Economic Forum Global Agenda Councils for Illicit Trade, Illicit Economy, and the Future of Electronics. At CSU, he is an FIIE (Faculty Institute for Inclusive Excellence) Fellow and a 2022 Best Teacher awardee. He has a cadre of on-campus students in his engineering departments, along with a larger contingent of online/remote graduate students researching in a wide variety of disciplines.

Dr. Art Villanueva is the chief AI/ML technology architect for Federal Strategic Programs at Dell Technologies. He has previously served as lead systems engineer for multiple high-profile programs, including billion-dollar initiatives. Art is an entrepreneur, having founded two renewable energy startups, one of which is now trading on NASDAQ; and an inventor with three patents. He has a Doctor of Engineering (D.Eng.) degree in Systems Engineering from Colorado State University, obtained his master's degree in Architecture-based Enterprise Systems Engineering from UCSD, and received his BS in Applied Mathematics with a Specialization in Computing from UCLA.