Utilizing AutoPhrase on Computer Science papers over time

Cameron Brody
crbrody@ucsd.edu

Jason Lin
jylin@ucsd.edu

James Yu
jjy002@ucsd.edu

Abstract

Phrase mining is a useful tool to extract quality phrases from large text corpora. Previous work on this topic, such as AutoPhrase, demonstrates its effectiveness against baseline methods by using precision-recall as a metric. Our goal is to extend this work by analyzing how AutoPhrase phrases change over time, as well as how phrases are connected with each other by using network visualizations. This will be done through exploratory data analysis, along with a classification model utilizing individual phrases to predict a specific year range.

1 Introduction

Phrase mining is the process of utilizing automated programs for extracting important and high-quality phrases from bodies of text. These phrases can be used in a variety of ways, from extracting major ideas from customer reviews or key points from a scientific paper. However, phrase mining has historically been done with complicated linguistic analyzers trained on specific data, meaning that it is difficult to expand to a larger scope without significant additional human effort. As a way to mine phrases in an expandable way, in any language or domain, AutoPhrase was created. With AutoPhrase, it is possible to input any text corpora without the need for human labels, allowing for much faster extraction of phrases in a variety of documents.

With that in mind, we utilized AutoPhrase to extract the phrases from a database of 3,079,007 computer science research papers aggregated from 1950 to 2017. With this, we can trace the evolution of key ideas through the history of computer science, as well as find which ideas were most common in what years. Additionally, we used the extracted phrases as data to construct a classification model for finding what year a paper belongs to based on its key phrases as a way of showing how strong the connections are between ideas and time.

2 Methods

2.1 Data gathering and processing for DBLP v10 + v13 datasets

Our initial goal was to gather data on Computer Science papers over time, looking at titles, abstracts, and paper contents. However, we realized that gathering and working with entire paper contents would result in much larger and messier data, while possibly not benefitting the results of AutoPhrase and our model. As a result, we chose to focus on the DBLP dataset (link). We chose this dataset as it contains a large amount of papers (3 million+) with information on each paper's title, abstract, and publication year. There are 13 versions of the dataset, but ultimately we chose to focus on the v10 dataset.

Our initial data processing was done on both the DBLP v10 and v13 datasets. The v13 is the latest version of the DBLP dataset from AMiner, released in May of 2021 with over 5 million papers. It contains all of the information previously specified, but it also includes keywords for each paper. We thought this would be beneficial as it allows for a point of comparison against the phrases we

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

would extract in the future by utilizing AutoPhrase. However, the v13 dataset had many issues with formatting that caused issues when trying to process it. The entire dataset is contained in a .json file that is too large to store in memory, so we had to process it line-by-line. However, the information for each paper is not contained on a single line–rather, it is spread out across multiple lines. This results in issues while processing each paper, as there are formatting issues that need to be resolved with many different cases.

The DBLP v10 dataset has fewer papers compared to v13 as it was released in 2017, but it still has information on 3 million+ papers. Additionally, it is much easier to work with as the information for each paper is stored in a single line. We created a function that goes through the dataset line-by-line and outputs the relevant information into .txt files. Our goal is to run AutoPhrase on the yearly aggregate of titles and abstracts, so we outputted .txt files for each year from 1950 to 2017. When processing the papers, we realized that there were papers with empty abstracts or invalid years. Thus, we chose to exclude any papers with empty abstracts and invalid years from the output .txt files. We specified invalid years as anything prior to 1950 and anything after 2017. In total, there were 530,394 papers with empty abstracts, and 82 papers with invalid years.

2.2 Exploratory data analysis for DBLP v10

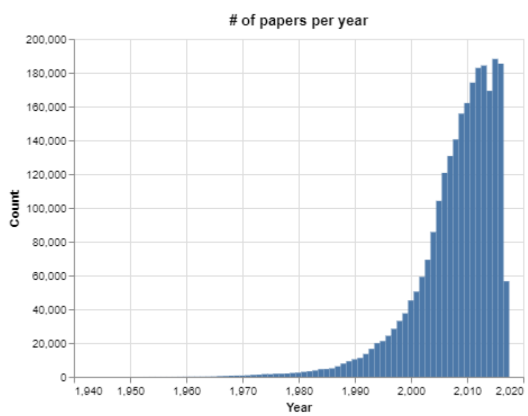

Figure 1: Document count for DBLP v10

The graph only includes papers that were included in our output .txt files. So the papers with empty abstracts or invalid years were not included. DBLP v10 contains 3,079,007 papers, but from our data processing steps, we filtered out 530,394 of the papers for having empty abstracts, and 82 of the papers for having irrelevant years (anything before 1950). Thus, this graph shows the distribution of the remaining 2,548,531 papers.

2.3 Running AutoPhrase

Running AutoPhrase only requires to have the input text data stored in a .txt file. As mentioned previously, we created a function that processes the DBLP v10 dataset and aggregates the titles and abstracts together in .txt files by year. However, when running AutoPhrase, it does need sufficient training data, meaning that if the input .txt file is too small, the results will be mostly incoherent. We found that the minimum file size is around 200-300 kilobytes, but it is not always consistent, as some smaller files were able to run without errors. Regardless, it is better to have larger files.

When running AutoPhrase on individual years, we found that the data from 1950-1967 was too small to run AutoPhrase on. As a result, we decided to group years in 5 year segments, with the exception of the first and last year segments in the dataset. The first year range, 1950-1959, is grouped into a 10 year range as there are not many papers in these years, as seen in Figure 1. The last year range, 2015-2017, only spans 3 years as the dataset was published in late 2017 and thus does not have any more papers.

3 Results

3.1 Phrase mining results on a single year

Table 1: AutoPhrase results on 2010.txt

Phrase Quality	Phrase
0.9659395508	game theory
0.9657724287	cognitive radio
0.9655697870	fourier transform
0.9654583931	reverse engineering
0.9650730867	knowledge base
0.9646135631	belief propagation
0.9641688928	remote sensing
0.9639178109	random walk
0.9635160488	shortest paths
...	...

When running AutoPhrase on a single year's .txt (containing all of its papers' titles + abstracts), we get an output of the phrases, along with their associated phrase qualities. Phrase quality ranges from 0.0-1.0, where 1.0 is the highest quality. We can typically associate high-quality phrases with single-word phrases with a score above 0.8 and multi-word phrases with a score above 0.5. These phrases provide insight into the various topics covered in just a single year of published Computer Science papers.

From the DBLP v10 dataset, we processed and outputted information on papers from 1950 to 2017, creating .txt files for each year. AutoPhrase was run on each of these files, giving us the output.

3.2 Phrase mining results on all papers

Table 2: AutoPhrase results on all years

Phrase Quality	Phrase
0.9812799074	video surveillance
0.9810695393	matrix multiplication
0.9808562642	antenna array
0.9807593231	nvidia cuda
0.9805699993	constraint satisfaction
0.9800784969	microsoft excel
0.9799606863	latin america
0.9798034992	template matching
0.9794710826	trapdoor permutations
...	...

We also ran AutoPhrase on an aggregate .txt file containing the titles and abstracts of all of the papers in the dataset from 1950-2017. These results contain information on the dataset overall, but are not useful for creating a model since we cannot associate each phrase with a year. It is possible, but would require additional work by going through each of the input papers and checking for each phrase. However, this is not necessary due to our AutoPhrase runs of the aggregated papers by year from the above section.

3.3 Consolidating phrase mining results

Table 3: Unique phrases overall

Phrase Quality	Phrase	Year
0.8901666667	time sharing	1968
0.61	real time	1970
0.9641666667	pattern recognition	1972
0.8661666667	data base	1972
0.8501666667	programming languages	1972
0.6201666667	computer science	1972
...	...	...
0.604091	reality vr	2017
0.602875	limited training	2017
0.602173	public datasets	2017

Table 4: Unique phrases by year

Phrase Quality	Phrase	Year
...	...	...
0.964167	pattern recognition	1972
0.866167	data base	1972
0.850167	programming languages	1972
0.620167	computer science	1972
0.981000	pattern recognition	1973
0.854000	database	1973
0.808500	linear programming	1973
...	...	...

After running AutoPhrase on each of the years (Section 3.1), we consolidated all of the results into a single .csv file. There were two approaches we took to this. The first looking at the unique phrases overall, meaning the first instance of the phrase is the only one included in the output file. So if a phrase such as 'image processing' were to appear in 1981, and also in 1982, only the instance in 1981 would be included in the output file.

The second approach looks at the unique phrases by year. Rather than only including the first instance of each phrase, duplicate phrases can appear across years. This allows us to see when phrases first appear, as well as the subsequent years they appear in.

3.4 Phrasal segmentation results

Phrasal segmentation takes in a .txt file and marks any mined phrases with phrase markers.

<phrase>Modular exponentiation</phrase> is a <phrase>cornerstone</phrase>
operation to several <phrase>public-key cryptography</phrase> systems such as the
<phrase>RSA</phrase> It is performed using successive modular multiplications The
latter is time consuming for large operands.
Accelerating <phrase>public-key cryptography</phrase> software or hardware needs
reducing the total number of <phrase>modular multiplication</phrase> needed.

Figure 2: Example phrasal segmentation results

Figure 2 shows an example of what phrasal segmentation does to text data. Any mined phrases with be marked with phrase markers. The phrase markers and phrases are highlighted in this screenshot for clarity. By processing the phrasal segmentation results, we can extract the marked phrases and group them together. This allows us to see the phrases mined by AutoPhrase on a per-paper level.

For instance, with the example, if we consider it the text for a single paper, we can see that it contains the phrases: modular exponentiation, cornerstone, public-key cryptography, and RSA.

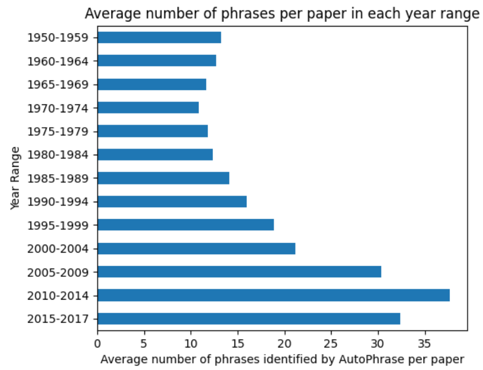

Figure 3: Bar chart of average phrases identified over time

This chart visualizes the average number of phrases identified by AutoPhrase for each year range. From the phrasal segmentation results, we are able to identify the phrases contained in each paper in the dataset. We can then take the average number of phrases identified per paper, and graph that information.

Here, we can see that the number of phrases per paper changes drastically depending on the year range. This can be due to factors like average length of input papers for that year range, but could also be dependent upon the range of phrases displayed within that range. A year range with more phrase variety could have less phrases show up per paper due to the lower average scores of the phrases causing them to be excluded from our high-quality phrase list.

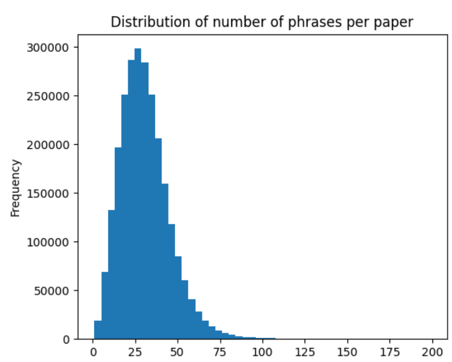

Figure 4: Histogram of number of phrases identified across entire dataset

This histogram shows the distribution of the number of phrases identified across the entire DBLP dataset. Overall, the number of phrases in a paper can vary widely, but the vast majority lie between 15 and 50 phrases.

3.5 Direct phrase matching and phrase similarity

Table 5: Direct phrase matching for ’convolutional neural networks’

Phrase Quality	Phrase	Year
0.865809	convolutional neural networks	2012
0.915629	convolutional neural networks	2013
0.937014	convolutional neural networks	2014
0.931728	convolutional neural networks	2015
0.917273	convolutional neural networks	2016
0.904261	convolutional neural networks	2017

Table 6: Phrase similarity for ’convolutional neural networks’ (Using unique phrases overall)

Phrase Quality	Phrase	Year	Distance
0.865809	convolutional neural networks	2012	0.0
0.900172	convolutional neural network	2013	1.0
0.839879	convolution neural network	2016	3.0
0.918423	convolutional neural networks cnn	2015	4.0
0.915458	convolutional neural network cnn	2014	4.0
0.889687	deep convolutional neural networks	2014	5.0
...	...	...	...

Using the unique phrases by year file, we can read in the .csv file using Pandas and perform various operations. For example, when looking at value counts, we can see popular phrases that show up many times, such as ’natural language’, ’data structures’, ’artificial intelligence.’ We can use Pandas to check for direct matches of a phrase, such as checking the rows that have the phrase ’image processing’. When doing so, the phrase first appears in our dataset in 1981 and has appeared in every year since, all the way until 2017.

Although phrase matching allows for us to directly find a phrase and the years in which it appears, it does not account for potential misspelling or non-direct matches. For example, if we tried to match for ’convolutional neural networks’ but the dataset only contained ’convolutional neural network’ (not plural).

We utilized the Levenshtein package to measure the Levenshtein distance between strings. This allows for us to find phrases in the dataframe that may not be exact matches, but are similar enough to warrant further analysis. When looking for the phrase ’convolutional neural networks’, there is a direct match in the dataframe, but there are also other phrases that are extremely similar, such as ’convolutional neural network’ and ’convolutional networks’. This approach looking at phrase similarity allows for us to find the most similar phrases to the input phrase, without having to worry about having a direct match in the dataframe. This idea can be pursued further to consolidate phrases within the AutoPhrase results.

This can also be used as a baseline method to classify an input paper’s year. For example, if we take in a paper’s title and abstract, we can extract the phrases within it. Perhaps by using n-grams or by looking for similar phrases within the dataframe, since AutoPhrase cannot run on too little data. Then we can use phrase similarity to find the similar phrases. So, if a paper was about convolutional neural networks, we could base our prediction off of our AutoPhrase results, classifying it as being published in sometime from 2012-2017 (or in a year after 2017).

3.6 Highest Quality Phrases over time

Table 7: Highest Quality Phrases across year ranges

1950-1959	1960-1964	1965-1969	1970-1974	1975-1979	1980-1984	1985-1989	1990-1994	1995-1999	2000-2004	2005-2009	2010-2014	2015-2017
operations research	tunnel diode	information retrieval	dynamic programming	fault tolerant	fault tolerance	logic programming	image segmentation	machine learning	heat transfer	block ciphers	option pricing	home automation
memory	differential equations	turing machine	markov chain	predicate calculus	human factors	pattern recognition	resource allocation	load balancing	belief propagation	microphone array	blind deconvolution	option pricing
magnetic	high speed	integer programming	question answering	linear programming	packet switching	petri net	petri net	temporal logic	stock market	hamming distance	laser scanner	rician fading
binary	data processing	data processing	programming languages	image processing	knowledge base	shortest path	character recognition	dynamic programming	congestion avoidance	wiener filter	superposition coding	voltage regulator
data	retrieval	automata theory	computational complexity	structured programming	dynamic programming	user interface	transaction processing	sequent calculus	kalman filters	copyright protection	moral hazard	buck converter
high	tunnel	dynamic programming	information retrieval	floating point	virtual memory	neural network	virtual reality	resource management	pattern recognition	blood pressure	brightness temperature	cooperative jamming
machine	amplifier	boolean function	feature extraction	question answering	turing machines	path planning	fourier transform	vector quantization vq	hamming distance	cellular automata ca	persistent homology	molecular docking
model	modulation	partial differential equations	floating point	dynamic programming	markov chain	load balancing	deductive databases	reverse engineering	random walks	transitive closure	associative memories	viral marketing
rate	digital	context free	integer programming	feature extraction	knowledge representation	image processing	modal logic	gaussian elimination	cellular phone	life sciences	buck converter	semidefinite relaxation
probability	design	differential equations	fault tolerant	transitive closure	petri nets	relational algebra	information retrieval	knowledge representation	stream cipher	spectral subtraction	preventive maintenance	mutual exclusion

We looked at the top 10 quality phrases for our year groups to see how AutoPhrase's results differed across years. Taking a glance at these example phrases will help us determine if AutoPhrase's quality phrases would serve as good predictors of a year. What is immediately obvious is that the first category consisting of papers with years from 1950-1959 consists of much simpler phrases. This category has the most single word phrases in their top 10 and their phrases illustrate broad concepts in computer science. This is promising as early computer science papers would deal with more basic concepts and could be a good predictor of year. This trend is relatively followed as earlier papers do contain phrases essential to the basics of computer science such as data processing and information retrieval while papers written with later years contain more high level concepts such as vector quantization and more proper nouns like Rician fading. There are some other aspects that stand out when looking at table 8. The phrase dynamic programming appears in the top 10 of many year groups along with other phrases like information retrieval and feature extraction. The fact that AutoPhrase picks many high quality phrases that are not useful for discriminating year groups could lead to AutoPhrase's quality phrases being noisy data when trying to use for prediction. Another interesting factor is that the year category 2005-2009 contains a variety of phrases relating to biology such as cellular automata, life sciences and blood pressure. This could possibly be due to computer science as a field expanding into other disciplines once the foundations of computer science had been established. This could explain the appearance of many seemingly random phrases within later years that appear to have very little to do with the field of computer science.

3.7 Most popular phrases over time

Table 8: Most popular multi-word phrases across year ranges

1950-1959	1960-1964	1965-1969	1970-1974	1975-1979	1980-1984	1985-1989	1990-1994	1995-1999	2000-2004	2005-2009	2010-2014	2015-2017
operations research (82)	pattern recognition (27)	sequential machines (85)	pattern recognition (165)	natural language (253)	natural language (494)	expert systems (782)	neural network (2504)	neural network (4977)	neural network (6001)	web services (12672)	cloud computing (16170)	machine learning (11254)
gaussian noise (16)	regular expressions (22)	pattern recognition (75)	linear programming (122)	pattern recognition (132)	signal processing (268)	natural language (770)	natural language (1089)	genetic algorithm (1700)	data mining (4901)	neural network (12314)	machine learning (14046)	big data (10885)
differential equation (12)	differential equations (21)	linear programming (71)	sequential machines (82)	computer graphics (128)	dynamic programming (204)	programming language (509)	expert systems (832)	image processing (1663)	web services (3543)	data mining (9980)	wireless sensor networks (12345)	social media (9504)
dynamic programming (8)	linear programming (19)	analog computer (58)	computer graphics (72)	linear programming (106)	pattern recognition (192)	user interface (495)	image processing (827)	software engineering (1430)	software engineering (3188)	wireless sensor networks (9382)	neural network (11381)	cloud computing (8373)
standard model (8)	sequential circuits (15)	sequential machine (54)	dynamic programming (69)	problem solving (104)	linear programming (174)	artificial intelligence (398)	distributed systems (799)	distributed systems (1414)	genetic algorithm (3115)	genetic algorithm (8088)	data mining (11235)	power consumption (6124)

By processing the phrasal segmentation results, we can obtain the counts of each phrase in the input data. We specifically focused on the most frequent multi-word phrases across each year range in order to identify the most popular Computer Science topics in each period. In the early years, there is a large focus on pattern recognition, as it is in the top 5 in all of the year ranges from 1960-1984. Over time, this changes, with topics such as neural networks and machine learning becoming more prominent.

3.8 Phrase network visualization

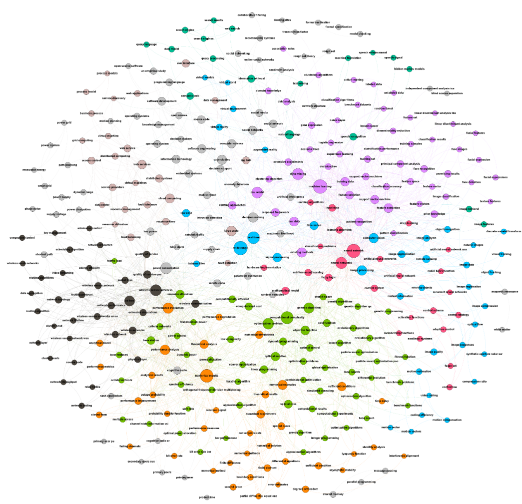

Figure 5: Network visualization

Higher-quality, zoomable image can be found here. This graph was created using the Gephi application after processing AutoPhrase's phrasal segmentation results on the DBLP v10 dataset.

This network visualizes the relationship between phrases for all papers in the DBLP v10 dataset (across all years). Phrases with more more occurrences in the dataset are represented by larger nodes in the network. Nodes are connected based on their connections in the paper. The phrasal segmentation results allowed us to extract the phrases identified for each individual paper in the dataset. With this, we could calculate the number of connections each phrase had with each other. For example, if 'neural network' and 'machine learning' are in the same paper, we would count that as 1 connection. With more connections across papers, edges between nodes have a larger weight.

Node colors are determined by modularity, so nodes with stronger edges to each other will be grouped together. For instance, with the purple nodes, 'machine learning' is the largest node, and we see other related nodes to that topic, such as 'decision trees', 'support vector machines', etc.

3.9 Phrase network by year range

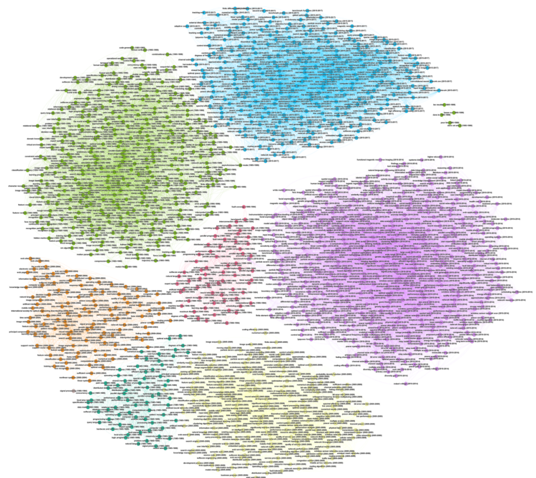

Figure 6: Yearly network visualization

Higher-quality, zoomable image can be found here.

This network isolates phrase relationships to their year range, providing insight into the most popular and connected phrases in each year range. The nodes colors are based solely on the year range of the phrase, rather than modularity. One fact to take into account is that the number of papers is much higher is recent years, so the frequency of phrases and their connections is much higher compared to earlier years. Steps were taken to normalize this difference across each year range and to only display the strongest and most meaningful relationships, but the number of nodes for each year range is not exactly equal. Ultimately, the purpose of this network is to provide a more intuitive understanding of phrase connections in relation to time.

3.10 Classification model

One of our initial goals for this project was to create a classifier to predict the year of a random Computer Science paper in order to demonstrate how distinct phrases contained within certain years have the capability to identify what year of the input paper. For this, we attempted multiple types of models including a Jaccard-based predictor, a predictor using phrase overlap between years, as well as trained models using one-hot encoding. However, we were only able to succesfully create a model by utilizing the phrase mining results, which only takes a single phrase as the input.

From creating a train-test split on the phrase mining results and training the model, we were able to achieve a 35.6% accuracy on the test set. However, this is relative to the baseline 35.3% accuracy when predicting the most common year range of the test set. So, while our model performs slightly better than the baseline, it is not great. Ultimately, we realized that using single phrases to predict what year range a paper belongs to is not the most reliable.

We hoped to utilize the phrasal segmentation results to build a model, but were unsuccessful in doing so. We believed that this could be a more reliable dataset to use, as it takes into account multiple phrases for a single row. This would also make more sense in respect to any other input test papers, as papers will typically contain multiple phrases in them, not just one.

4 Conclusion

After processing and exploring the DBLP v10 dataset, we were able to utilize both functions of AutoPhrase (phrase mining and phrasal segmentation) to extract meaningful data and explore the relationships between phrases further. We identified the change in phrases over time by looking at the most popular phrases for each year range. We analyzed the relationship between phrases on a per-paper level, utilizing the segmentation results, in order to create a network visualization. We analyzed this relationship in respect to time, visualizing the network of phrases for each year range. We created a classification model in order to predict the year range of a paper based on its phrases.

5 Future Work

Additional work can be done to improve the classification model idea. The proposed idea was to be able to pass in any random input paper title and abstract and obtain a prediction of the specific year. Perhaps by utilizing the phrasal segmentation results to train the model, it may be possible to revert back to making predictions to specific years, rather than the defined year ranges.

It would be interesting to explore an evolving network animation that starts with the first year in the dataset, showing all of the phrase relationships, then changes as we go through each year. This may not be possible directly in Gephi's software, but it could be done by creating separate graphs and maintaining certain color schemes. Additionally, exploring single-word phrases alongside the multi-word phrases could be interesting as well. It may require additional filtering of words to remove any meaningless phrases.