1. **Code:**Please include your code in the form of a Github repo link in a footnote somewhere in your synopsis. If you make changes to the code after the January 3rd synopsis deadline, you should push these to Github.
2. **Synopsis content:**The synopsis should - at time of submission - include implementation results. I.e. it cannot be simply a study plan/design, although you may talk about "next steps". If you make further progress after January 3rd and come prepared to discuss these results in the oral presentation, that's great (it will likely help you anticipate some of our questions, which may have a positive impact on your grade). But the synopsis should still stand up on its own.

The final project is to either build a natural language processing system, or apply one for some task. The project must use or develop a dataset, and report empirical results or analyses with the dataset. It may use machine learning or rule-based approaches. It may use any type of open-source or widely available software.You can choose to emphasize:

* Implementing and developing algorithms and features.
* Defining a new linguistic / text analysis task, and tackling it with off-the-shelf NLP software.
* Collect and explore a new textual dataset to address research hypotheses about it.

Different projects will have different balances of these three things.

The key requirement is to investigate, analyze, and come to research findings about new methods, or insights about previously existing methods.

The synopsis can be done as an individual or group assignment. If it is done as a group assignment, the final product must  
(i) form a coherent text and (ii) be organized so that it is possible to make individual assessments of the students contributing.  
In other words, the contribution of each individual student to the whole assignment must be clearly delineated (excluding the introduction, conclusion and bibliography).  
A maximum of three students can take part in a group assignment.

* Length of individual synopsis: 4-7 standard pages (not including code and figures)
* Length of synopsis for 2 students: 8-14 standard pages (not including code and figures)
* Length of synopsis for 3 students: 12-21 standard pages (not including code and figures)

The final synopsis  describes your project and final results. Conceptually, it should include the content of both the proposal and progress report, though they will be changed. The final report describes and motivates the problem, places it in context of related work, describes the dataset and your approach, and reports results with discussion and thoughts for future work.

Here is a sample outline for your final report. There are different possible ways to structure it (for example, if you can, you can weave related work into the other sections), but we suggest you follow this outline unless you have substantial prior experience writing technical reports and research papers.

* **Abstract:** summarize the main components of your work in one paragraph (no more than 5 sentences). What problem are you solving? What is the key to your approach? What results did you achieve? Your abstract should draw the reader in and interest them in reading the rest of your paper to understand the details of your work.
* **Introduction:** explain the problem, motivate it (why is it important?), and briefly describe your approach. State a research question that your project seeks to answer: what are you trying to learn from this research project? You may also report some of your results without discussing the details of your method.
* **Data:** Describe the dataset that you are using.
* **Method:** Describe your approach to handling the problem. This should should include any models you used and any modeling assumptions you made. If you’ve developed new models for this project, you may even want to split a description/analysis of your models into its own section.
* **Results:** Describe the experiments you ran and identify your baseline method(s). Include the results you achieved with the various methods you are comparing making. This section will probably also include some figures that succinctly summarize your results. Analyze your results (including your models). If you did exploratory analysis or a significant amount of feature engineering, your analysis may merit its own section. After reading this section (and your dataset and methods), an interested reader should be able to duplicate your experiments and results.
* **Discussion and Future Work:** discuss any implications of your analysis for the problem as a whole, and what are the next steps for future work. Any other concluding remarks should go here.

A Matter of Tech Security

Predicting Trends from Podcasts

# Abstract

The podcast Security Now has been running every week since mid-2005. Here it is used as a novel source of NLP data. Using LDA, transcripts of the podcasts are used to investigate how topics on tech security change over time. We find that

# Introduction

explain the problem, motivate it (why is it important?), and briefly describe your approach. State a research question that your project seeks to answer: what are you trying to learn from this research project? You may also report some of your results without discussing the details of your method.

Since the introduction of the General Data Protection Regulation (GDPR) in May 2018, awareness of personal data protection has risen substantially all across Europe. The GDPR helped introduce the subject of internet security to the casual user

The personal computer was introduced to the masses in the early 1990’s (‘Personal computer’, 2019) and the share of personal data we put online has only continued to grow since then. This brings up the issue of how to best protect our data and what safety rules to follow when we “travel online”. And while some still think their birthday is a good password for anything from social media accounts to web banking, many are beginning to see the necessity of properly protecting their data.

Here we are investigating whether changes in trends are observable over time in wide field of personal computing. In addition, we are interested in finding out if any of these trends or “topics” can be said to have been predicted preemptively by a podcast featuring two experts in the field.

## Quantitative Text Analysis with Topic Modelling

The aim of *quantitative* text analysis, as opposed to traditional *qualitative* text analysis, is to make *statistical* inferences from *large text corpora*. From this description, it is clear that such inferences are better suited for computers than humans. The first computerized text analysis program “The General Inquirer” was invented at Harvard in 1966 (Stone, Dunphy, & Smith, 1966). Around 1990 the term *Latent Semantic Indexing* (LSI) began making its way into the scientific literature on language and information sciences. The power of this new method lay in its ability to reduce “noise” in text retrieval by assuming an underlying (or “latent”) semantic word structure. ~~In other words search inquiries needed to be less specific, as documents were sought after in a context manner rather than by direct word-matching~~ (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990; Dumais, Letsche, Littman, & Kandauer, 1997).

In 2003, Latent Dirichlet allocation (LDA) was introduced by (Blei, Ng, & Jordan, 2003). The model followed traditions from LSI but . The name

Since then a wealth of computerized

Now, the combination of computers with massive computing power and the increasing digitalization of data has caused a change in the way text analysis is possible (Mehl, 2006).

Topic modelling is a powerful tool used in the field of quantitative text analysis.

Latent: topics only emerge after analysis

Dirichlet: after Peter Gustav Lejeune Dirichlet, a probability distribution consisting of a range of probability distributions (like words in topics in docs)

Allocation:

Probability distributions rather than strict wod frequencies (<https://www.youtube.com/watch?v=DWJYZq_fQ2A>)

LSI (also known as Latent Semantic Analysis, LSA) learns latent topics by performing a matrix decomposition (SVD) on the term-document matrix.  
  
LDA is a generative probabilistic model, that assumes a Dirichlet prior over the latent topics.  
  
In practice, LSI is much faster to train than LDA, but has lower accuracy.

## Case Study: Security Now Podcast

The podcast “Security Now” (SN) has been running continuously since 2005. Its two hosts, Steve Gibson and Leo Laporte, in their own words “*spend somewhat shy of two hours each week to discuss important issues of personal computer security.”* (‘GRC | Security Now! Episode Archive’, n.d.). Here we use transcripts of the podcasts to train several LDA models,

# Materials and Methods

## Data statement

Data consisted of transcripts of 730 “Security Now” podcast episodes, scraped from the public website www.grc.com/securitynow.htm. Data was gathered strictly for academic research purposes. Speaker demographic was predominantly homogenic (white, male, American, 30+) and consistent throughout the dataset. The texts were of semi-unstructured, conversational nature and the topic range was narrow; each episode discusses issues of personal computer security, but due to the format are sometimes more private in nature. Data was scraped for the period August 2015 - October 2019.

## Github Repository

All scripts used for analysis can be found in the github repository <https://github.com/isalykke/NLPexam>

## Pipeline

Below we describe the steps used in finding and analyzing the dataset:

1. Data was scraped from the website [www.grc.com/securitynow.htm](http://www.grc.com/securitynow.htm) using the script “*security\_now\_scraper\_beautifulsoup.py*”. The script utilizes the **python** package **beautiful soup** (Guido & Drake Jr, 2016; Richardson, 2007).
2. All files were merged and metadata was extracted using the “*txts\_to\_csvs.Rmd*” script. This step was done in **R** (R Core Team, 2019) and implemented in **RStudio** (RStudio Team, 2019).
3. Text cleanup, word removal and lemmatization was done in Visual Studio Codeusing the python package **nltk** (Bird, Klein, & Loper, 2009) – SPECIFY THE SCRIPT
4. LDA topic modelling was similarly done with python utilizing the **genism** (Radim & Sojka, 2010) package – SPECIFY THE SCRIPT
5. Google data for comparisons was downloaded from <https://trends.google.com/trends/>. Search parameters were “USA”, “2004-present”, “All categories” and “Websearch”.
6. Finally, visualizations of the models were done in both python and R, using **ggplot** (Wickham, 2016) and **matplotlib** (Hunter, 2007).

# Results

Baseline method:

# Discussion

Number of topics

## Questions for further research

Bigram models

Comparing to non-informative words – do these also predict something?

# Conclusion

# References