

# Leveraging the Power of ChatGPT and Vector Databases in the FreeBSD Expert System

Yan-Hao Wang, AsiaBSDCon 2024

# Who Am I

My name is Yan-Hao Wang, a senior high student in Taiwan and FreeBSD Taiwan intern since 2022.


I've been involved in various tasks such as

1. Developing an online document/man-page editor.
2. Crafting tests for command utilities like `gunion(8)` and `printenv(1)`.
3. Translating FreeBSD documents.


# GitHub Repository

All codes have been uploaded to the **freebsd\_data** repository. The slide will also be put on it. If you're interested, you can access them there.


The screenshot shows the GitHub interface for the repository 'freebsd\_data'. At the top, there is a header with the repository name, a 'Public' badge, and interaction buttons for Pin, Unwatch (2), Fork (0), and Star (3). Below the header, there is a navigation bar with 'main' branch selected, '1 Branch', and '0 Tags'. A search bar 'Go to file' is present, along with a '+ Code' button. The main content area shows a commit by 'Wang-Yan-Hao' titled 'Refactor "example/retrieve/" codes' with commit hash 'fd47bd8' and '4 Commits'. Below the commit, a file 'example/retrieve' is listed with the same commit message and time. On the right side, there is an 'About' section with a gear icon, containing the text 'Script to extract documentation and man page data from FreeBSD.' and a 'Readme' link.

 **freebsd\_data** Public Pin Unwatch 2 Fork 0 Star 3

main 1 Branch 0 Tags  t + <> Code

 **Wang-Yan-Hao** Refactor "example/retrieve/" codes fd47bd8 · 17 hours ago 4 Commits

example/retrieve Refactor "example/retrieve/" codes 17 hours ago Readme

**About** 

Script to extract documentation and man page data from FreeBSD.

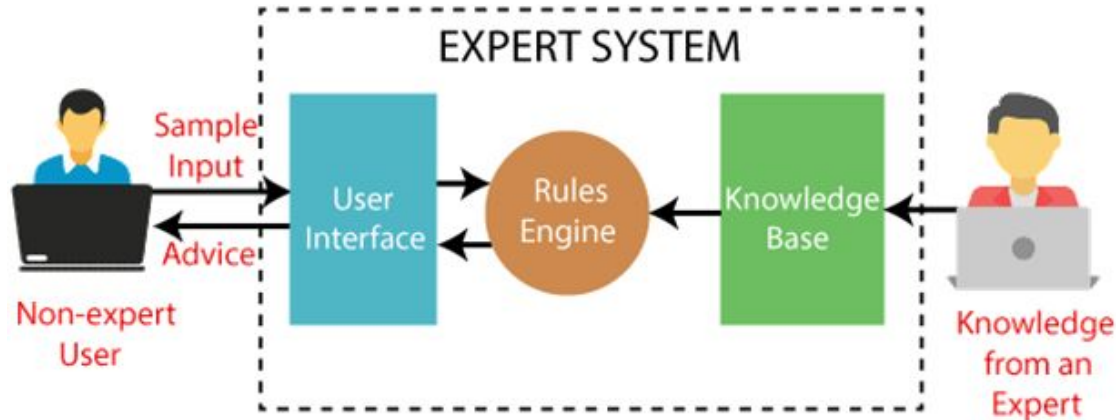
# Outline

1. Introduction of the Expert System
2. Introduction of ChatGPT
3. Development Process
  - a. Data Cleaning and Extraction
  - b. Embedded Model and Vector Database
  - c. Integration with ChatGPT
4. OpenAI GPTs as Potential Replacements
5. Summary

# Expert System

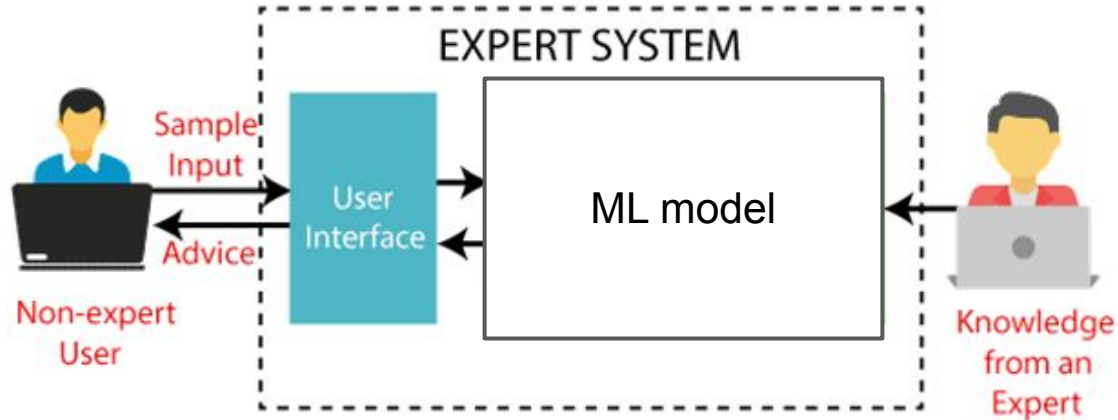
Expert system is a system that can answer user questions accurately in a specific domain. It consists of two parts

1. **Knowledge Base:** stores all the relevant information related to the domain of expertise.
2. **Rule Engine:** Contain some predefined rules by the data scientist. It processes the user's questions and applies rules to generate accurate responses.



# Expert System

Modern expert systems use **machine learning** to simulate the behavior or judgment of domain experts.



# ChatGPT

ChatGPT (Chat Generative Pre-trained Transformer) is a chatbot developed by OpenAI and launched on November 30, 2022. Based on a large language model (LLM).

# FreeBSD Expert System

There are multiple ways to build a FreeBSD expert system.

1. Train a new ML model with FreeBSD data.

No. I am not an ML expert and it costs a lot.



# FreeBSD Expert System

There are multiple ways to build a FreeBSD expert system.

1. Train a new ML model with FreeBSD data.

No. I am not an ML expert and it costs a lot.

2. Use the existing model such as ChatGPT.

But we definitely won't call ChatGPT a FreeBSD expert system.

# FreeBSD Expert System

ChatGPT uses amount of data for training. So he can answer problems in every domain though may not be correct. It's more like a **general-purpose system**.

# FreeBSD Expert System

ChatGPT uses amount of data for training. So he can answer problems in every domain though may not be correct. It's more like a **general-purpose system**.

The limitation of why ChatGPT can't be called a FreeBSD expert system

1. Chatgpt may tendency to hallucinate answers when asked about unfamiliar domains.
2. The data is not new enough (ChatGPT uses data before 2021 to train). So he can't answer the newest question.

# FreeBSD Expert System

There are two ways to handle the limitation.

1. **Fine-tune**. fine-tuning is a process that takes a model that has already been trained for one given task and then tunes or tweaks the model to make it perform a second similar task.

# FreeBSD Expert System

There are two ways to handle the limitation.

First way is, **fine-tune**. fine-tuning is a process that takes a model that has already been trained for one given task and then tunes or tweaks the model to make it perform a similar task.

OpenAI has supplied this API. For the open-source model, you should use Pytorch and TensorFlow to handle it.



OpenAI

<https://platform.openai.com>

## Fine-tuning

**Fine-tuning** lets you get much better quality results than prompts.

# FreeBSD Expert System

However, fine-tuning is still hard for AI-unfamiliar developer. And It also cost a lot.

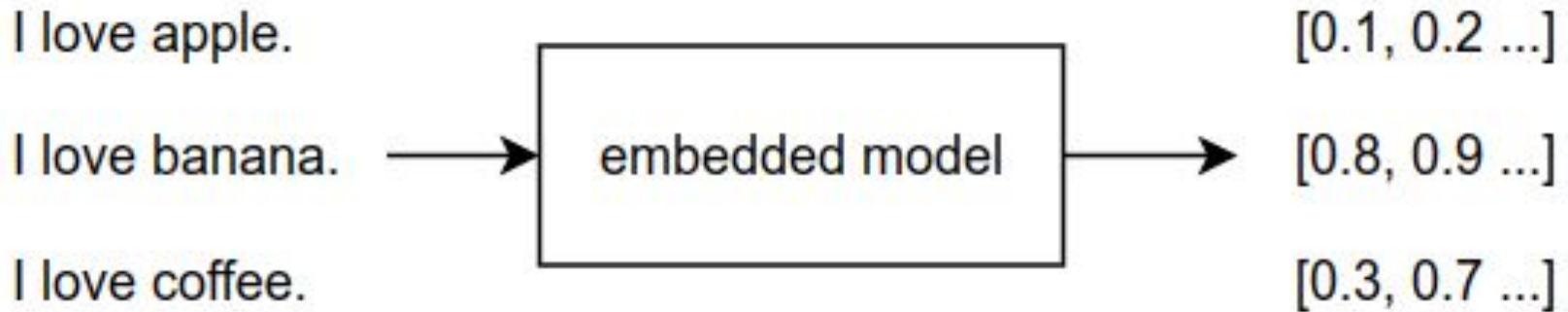
The second way is **Retrieval Augmented Generation (RAG)**. Basically, it is just like when you use ChatGPT, you can provide related info about your question, and it can provide a much more accurate response.

This is an acceptable way, so we will use the **embedded model** and **vector database** to achieve this.

# Embedded Model

It is a type of ML model used to convert input data, such as words or sentences, into numerical representations called **embedding vector** or **vector**.

These embeddings capture the semantic meaning or context of the input data in a continuous vector space. It can work on tasks such as text classification and sentiment analysis.



# Vector Database

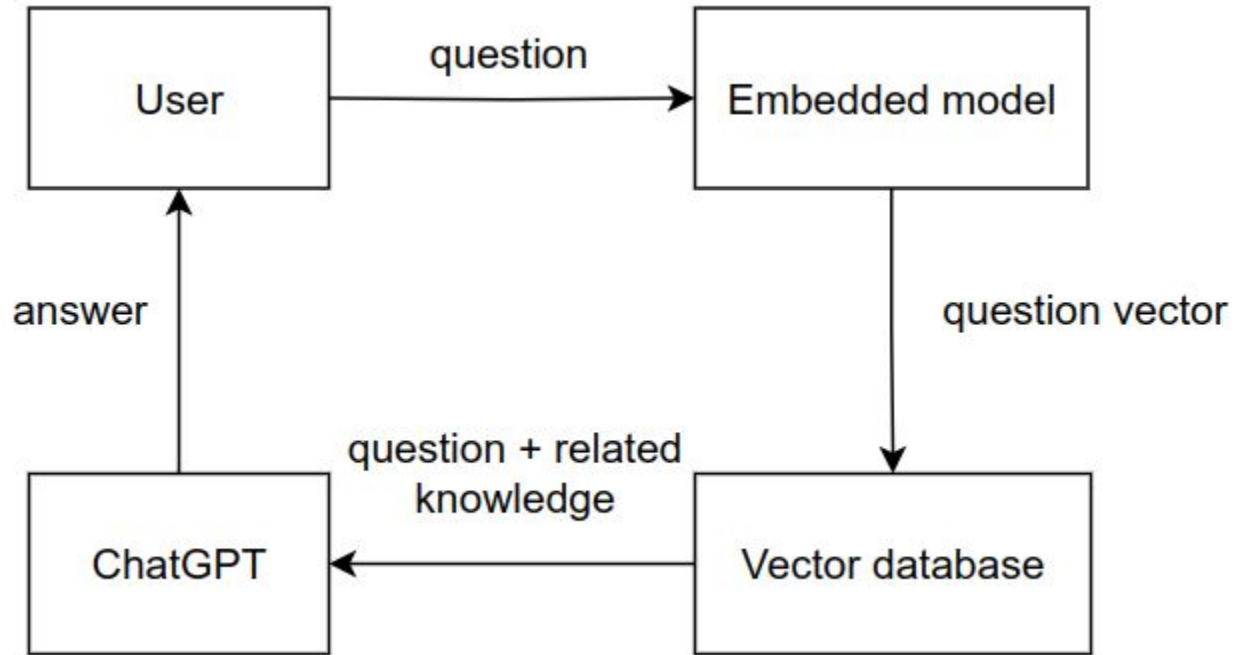
Vector databases are designed to store vectors efficiently. These databases employ various search algorithms to find the most similar vectors, such as t. Numerous open-source vector databases are available to choose from.





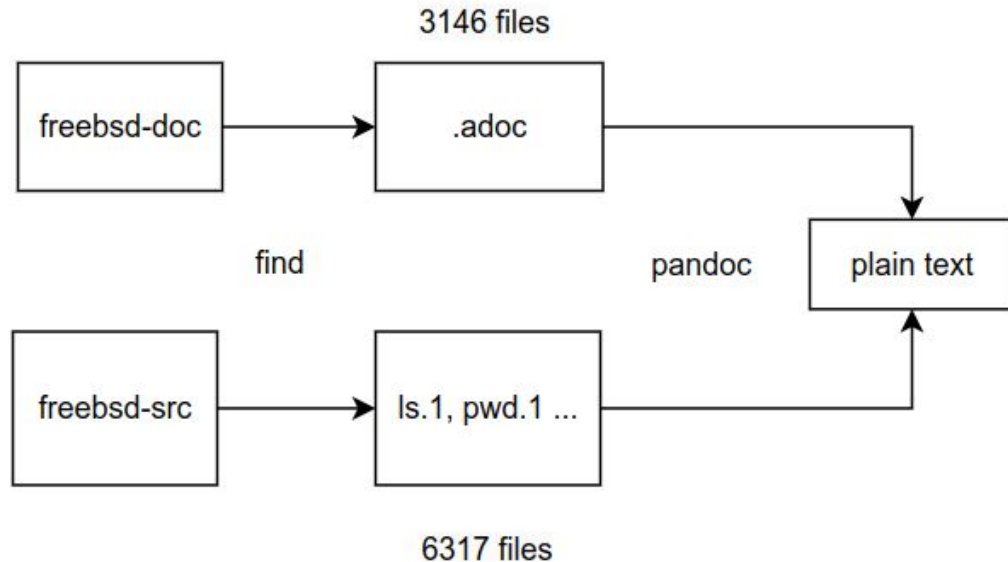
# Development Process

# Development - Architecture



# Development - Data Extraction

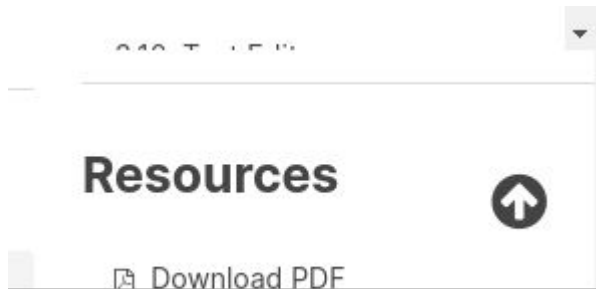
Use the simple find command to extract data. The data sources are very different, we need to convert it to plain text. We use “hs-pandoc” package to convert data.



# Development - Data Cleaning

Remove unrelated info, simple **find** command to remove the unrelated data.

```
# Remove irrelevant content in the text files
find "data/doc" -type f -name "*.txt" -exec sh -c 'awk "/^\[FreeBSD logo\]$/, /^♥ Don
find "data/doc" -type f -name "*.txt" -exec sh -c 'awk "/^trademarks$/,/^symbol\.$/ {
find "data/doc" -type f -name "*.txt" -exec sh -c 'awk "/^Resources$/,/^- Edit this p
find "data/doc" -type f -name "*.txt" -exec sh -c 'sed "/^--*$/d" "$1" > "$1.tmp" &&
```



Unrelated text

# Development - Data Cleaning

Actually, data cleaning is the most time-consuming step. **Data scientists spend 60% of their time cleaning data rather than creating insights.**

There are some tools that can help us clean the data.



OpenRefine



TRIFACTA



# Development - Embedded Model

OpenAI has embedded model API, there are multiple open source embedded models online too. In this project, we use the open source model (“gte-base”).

MODEL	~ PAGES PER DOLLAR
text-embedding-3-small	62,500
text-embedding-3-large	9,615
text-embedding-ada-002	12,500

OpenAI embedding model

Rank ▲	Model ▲	Model Size (GB) ▲
1	<a href="#">SFR-Embedding-Mistral</a>	14.22
2	<a href="#">voyage-lite-02-instruct</a>	
3	<a href="#">GritLM-7B</a>	14.48
4	<a href="#">e5-mistral-7b-instruct</a>	14.22

MTEB Leaderboard - From Hugging Face

# Development - Embedded Model

Welcome to FreeBSD! This handbook covers the installation and day to day use of FreeBSD 14.0-RELEASE and 13.2-RELEASE. This book is the result of ongoing work by many individuals. Some sections might be outdated. Those interested in helping to update and expand this document should send email to the FreeBSD documentation project mailing list.



vector 1 [0.2, 0.3 ... 2.3]



vector 2 [0.3, 0.6 ... 1.7]



vector 3 [0.9, 0.1 ... 3.1]

# Development - Embedded Model

We use “gte-base” as our model. Its model size is only 0.22 GB which my small GPU (NVIDIA GeForce GTX 1050 Ti) can handle it.

It takes only 590MB of GPU memory and 67 minutes to embed all the documents.

Model Name	Model		Sequence Length	Average (56)	Clustering (11)	Pair		
	Size (GB)	Dimension				Classification (3)	Reranking (4)	Retrieval (15)
<u>gte-large</u>	0.67	1024	512	<b>63.13</b>	46.84	85.00	59.13	52.22
<u>gte-base</u>	0.22	768	512	<b>62.39</b>	46.2	84.57	58.61	51.14



# Development - Embedded Model

```
Every 1.0s: nvidia-smi kola-Z590-GAMING-X: Mon Mar 18 17:49:25 2024
Mon Mar 18 17:49:25 2024
+-----+
| NVIDIA-SMI 545.23.08                Driver Version: 545.23.08      CUDA Version: 12.3     |
+-----+-----+-----+-----+-----+-----+
| GPU  Name                Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |      Memory-Usage | GPU-Util  Compute M. |
|====+=====+====+=====+=====+=====+
|  0   NVIDIA GeForce GTX 1050 Ti      On          | 00000000:01:00.0 On          |         N/A         |
| 60%   72C    P0              N/A / 72W | 1006MiB / 4096MiB |    97%    Default   |
|                                         |                      |         N/A         |
+-----+-----+-----+-----+-----+-----+
+-----+
| Processes: |
| GPU  GI   CI       PID   Type   Process name                      GPU Memory |
|   ID  ID   ID                   |                Usage                     |
|====+=====+=====+=====+=====+=====+
|  0   N/A N/A       2597   G     /usr/lib/xorg/Xorg                  218MiB |
|  0   N/A N/A       2731   G     /usr/bin/gnome-shell                 27MiB |
|  0   N/A N/A       9610   G     ...seed-version=20240315-130113.878000 38MiB |
|  0   N/A N/A      14301   G     ...,WinRetrieveSuggestionsOnlyOnDemand 61MiB |
|  0   N/A N/A      16140   G     ...sion_SpaceRendererForSitePerProcess 48MiB |
|  0   N/A N/A      30170   C     python                              590MiB |
+-----+-----+-----+-----+-----+-----+
+-----+
```

# Development - Embedded Model

```
(autopytorch) (base) kola:~/git-project/freebsd_data/example/retrieve/embedding$ python main.py
Found 8854 files
Using device: cuda
Token indices sequence length is longer than the specified maximum sequence length for this model (
985 > 512). Running this sequence through the model will result in indexing errors
Total chunks from all documents: 47623
Processing chunk 0
Processing chunk 10000
Processing chunk 20000
Processing chunk 30000
Processing chunk 40000
Time taken for tokenization and inference: 3958.3881 seconds
```

# Development - Embedded Model

There are multiple facts (hyperparameters) we can tune here. For example

1. The length of sentences.
2. What metadata should we leave?
3. What model should we use? Whether we need to tune the embedded model.

All these hyperparameters should be tried multiple times to get the best answer.

The answer will be different with different fields - NFL(No Free Lunch Theorems)

# Development - Vector database

As previously said, we have different vector databases.

But in our local test, we just use a **file** to store the vector and a simple **cosine similarity algorithm**. Because our data is not big (< 100 MB).



# Development - Query

Question: How to use the gunion command in FreeBSD?

Query result:

1. Man page of gunion
2. Man page of gunion
3. FreeBSD status report (A New GEOM Facility, gunion)
4. Unrelated info ...

# Development - Query

```
(autopytorch) (base) kola:~/git-project/freebsd_data/example/retrieve/query$ p  
ython query.py  
Current Question: How to use the gunion command in FreeBSD?  
Related Chunk Texts:  
Using device: cuda  
--- Top 1 Result ---  
Chunk Text: [CLS] - - - - - gunion ( 8 ) freebsd system  
manager's manual gunion ( 8 ) - - - - -  
name gunion co
```

TOP1

```
--- Top 2 Result ---  
Chunk Text: 8 ) manual page. at this time, gunion ( 8 ) is available only in 1  
4. 0. sponsor : netflix last modified on : june 3, 2022 by lorenzo salvadore l  
egal notices | © 1995 - 2024 the freebsd project all rights reserved. the mark
```

TOP2

```
--- Top 3 Result ---  
Chunk Text: [CLS] [ freebsd the power to serve ] donate to freebsd - home - ab  
out - introduction - features - privacy policy - projects - ports - get freebs  
d - release information - release engineering - documentation - faq - handbook  
- porter's handbook - developer's handbook - committer's guide - manual pages
```

TOP3

# Development - Integration with ChatGPT

So we need to host an embedded model and vector database and have an open API to let users use. Then integrate the API with ChatGPT

1. The first way is easy, we just write a Python code to use ChatGPT API and our API. But this is not friendly to normal users.

# Development - Integration with ChatGPT

So we need to host an embedded model and vector database and have an open API to let users use. Then integrate the API with ChatGPT

1. The first way is easy, we just write a Python code to use ChatGPT API and our API. But this is not friendly to normal users.
2. Develop **ChatGPT plugin**, ChatGPT plugin can let us set some API. While asking questions ChatGPT, it will call the API and get the response.

This is the best practice of our project, the user just needs to enable the plugin in ChatGPT.



# Development - Integration with ChatGPT

**You**

What is gunion in freebsd.

**ChatGPT**

It seems there might be a typo in your question. There is no specific reference to "gunion" in FreeBSD. However, if you meant "union" in the context of FreeBSD, it's a feature that allows a file system to appear to be both read-only and writable.



In FreeBSD, gunion is a control utility for managing the UNION GEOM class. The primary purpose of gunion is to facilitate tracking changes to a read-only disk on a writable disk. This is achieved by placing a writable disk over a read-only disk. When write requests occur, they are intercepted and stored on the writable disk. Read requests are first checked on the writable disk, and if the data is found there, it is returned. If not, the data is retrieved from the lower, read-only disk.

# OpenAI GPTs as Potential Replacements

GPTs was launched at November 2023. It provides an easy way to generate a custom GPT for any data you have. Which becomes a potential replacement for our project. We only need to upload the data from step 1 and there is a custom expert system.

On March 19, 2024, you will no longer be able to install new plugins or create new conversations with existing plugins.

## My GPTs



**Create a GPT** Beta

Customize a version of ChatGPT for a specific purpose

# Wiki Future Audiences

The idea is inspired by Wiki. They actually already have developed a plugin. But they also stopped the plan after the GPTs release.

*This timing also coincides with OpenAI's move away from the plugin marketplace for ChatGPT, and towards no/low-code customizable GPTs. This shift has made our plugin in its current form inaccessible to new users and largely redundant. While we could repurpose this functionality towards being a GPT, we don't believe we would learn significantly more beyond how to create a product within the OpenAI ecosystem.*

***Lessons learned, ChatGPT has not become the new information seeking paradigm (yet?).***

# Summary

<b>Solution</b>	RAG	GPTs (Custom GPT)	ChatGPT Plus (browse internet)
<b>Cost</b>	Medium ~ Hard	Small	Small
<b>Advantage</b>	<ul style="list-style-type: none"><li>● Privacy</li><li>● Flexibility</li></ul>	<ul style="list-style-type: none"><li>● Fast</li></ul>	<ul style="list-style-type: none"><li>● Fast</li></ul>
<b>Disadvantage</b>	<ul style="list-style-type: none"><li>● Cost</li></ul>	<ul style="list-style-type: none"><li>● Privacy</li><li>● Flexibility</li></ul>	<ul style="list-style-type: none"><li>● Data source are different</li><li>● Flexibility</li></ul>

# Summary

The significance of LLM is poised to exponentially increase in the future, marking a pivotal shift in our technological landscape.

While we may not complete the production process in its entirety. But it is a good thing to focus on any future trends and try to combine them with FreeBSD.

# Reference

- [What is an Expert System?](#)
- [Do data scientists spend 80% of their time cleaning data? Turns out, no?](#)
- [Wiki. Talk:Future Audiences](#)