Evaluation of a new Patent Search System

Target users:

In-house patent lawyers, and general public

Goals:

Evaluate the system before launch.

Make suggestions for improvements.

Tasks:

1) Gather information about the system requirements
2) Based upon the requirements, propose aspects that need testing/evaluating
3) Describe in detail the evaluation process and justify the proposed evaluation procedures
4) Based upon the selected evaluation methods, describe what results should be expected

---

Part 1: Identifying System Requirements

Patent information commonly refers to the information found in patent applications and granted patents. This information may include bibliographic data about the inventor and patent applicant or patent holder, a description of the claimed invention and related developments in the field of technology, and a list of claims indicating the scope of patent protection sought by the applicant. [1]

I interviewed a variety of target users in order to have a wider range of requirements, and obtained the following feedback:

- **General public (GP)**: they need to be able to check whether any patent has already been applied for or granted that is:
    - Similar to their invention
    - Used on related products or for related services,
    - and in active current use

  Meeting all three criteria will prevent the patent from being granted because it creates a likelihood of confusion. (there are other criteria for being granted a patent, but those are beyond the scope of using a patent search system)

- **Stakeholders (SH)**: It is important to make money out of the developing this new patent search system. This is more easily accomplished with appealing to a bigger volume of people. The search system needs to be user-friendly and handle large volumes of searches.

- **In-house lawyers (IHL)**: Searches cannot take a long time. The search system needs to give us results that are relevant. No relevant documents should be skipped.

This translates into the following **6 requirements**:

1) (GP): Recall is very important – have to find all relevant patents **(MUST)**
2) (GP): The system should recommend related search terms based on keyword analysis (for finding patents that are similar, and/or that are used in related services or products) **(SHOULD)**
3) (SH): user-friendly UI/GUI **(COULD)**
4) (SH): system must be capable of handling large volumes of simultaneous searches/users **(MUST)**
5) (IHL): search time must be low/reasonable **(SHOULD)**
6) (IHL): good speed and accuracy **(COULD)**

Part 2: Targets for Evaluation and Testing

Based upon the gathered requirements, the following **4 aspects** of the search system should be tested:

- **Recall and Precision:**
    - The Recall of the system is the proportion of relevant documents retrieved. This would cover requirement number 1).
    - The Precision of the system is the proportion of the retrieved documents that are relevant. This would cover the second part of requirement 6) that concerned "accuracy".
- **Efficiency (time lag)**: time interval between the time a request is made and the answer is given. This would cover requirement number 5) and the part of requirement number 6) that concerned "good speed".
- **Effort:** Effort refers to the amount of work needed by the user to find and identify the relevant documents. Our search system should require a low amount of effort by all types of users, whether they are professionals capable of writing very precise Boolean queries (such as in-house patent lawyers), or general public who might prefer queries using natural language. This would cover requirement 3): user-friendly UI.
- **Robustness** of the system: how to search system software behaves under extremely heavy load conditions. This would cover requirement 4): large volumes of simultaneous searches/users.

Regarding the **other requirements which are not being specifically tested**:

Requirement 2) is not being tested at this time since it concerns an interactive feature (recommendation of related search terms based upon the user's query) which was not built into the search system that is being tested. I highly recommend that the developers should consider adding this feature in one of the future releases of the search system software.

Part 3: Evaluation Methods

Recall and Precision

The basic metrics of any system are **Recall** and **Precision**. There is always a need to compromise when choosing whether to optimise Recall or Precision because they have an inverse relationship between them so it is not possible to optimize both at the same time.

$$Precision = \frac{\text{number of relevant documents retrieved}}{\text{number of documents retrieved}}$$

$$Recall = \frac{\text{number of relevant documents retrieved}}{\text{number of relevant documents}}$$

Why do we say that precision and Recall have an inverse relationship?

If we want a high value of recall in our search results (comprehensive retrieval), then we must include synonyms, related terms, abbreviations, broad or general terms, etc. for each query term. Precision will then be lower since a higher number of documents will be included in the search results.

Synonyms may not be exact synonyms, which increases the probability of retrieving irrelevant documents. Broader terms may retrieve documents that do not discuss the narrower search topic. Hence, a high Recall leads to a lower Precision.

A high precision will retrieve a smaller amount of documents with the goal that all of the documents retrieved (or close to all) will be relevant. This means that, in order to be "certain" that all our retrieved documents are relevant, there are many documents that are not retrieved which could potentially have some relevance. Hence, we get a lower Recall with a higher Precision.

Which is more important in a Patent Search System – Recall or Precision?

In the case of Patent Searches our key metric is Recall. Time matters less than being able to see every relevant document. Although we cannot afford to miss a single relevant document, we should also not completely sacrifice Precision since if we have a huge amount of documents retrieved, this will lead to a lot of time being wasted perusing through "false positive" results. A balance should be struck: we should favour recall but not completely sacrifice precision.

Measuring Recall and Precision:

We would need to have volunteers to help judge which documents are truly relevant for given queries (relevance judgment). Ideally the pool of volunteers would be as representative as possible of the actual target end users.

Steps for testing:

1. We create a Test Collection, e.g. a sample of documents/patents from the patent database that we are searching with the new search system *
2. We create queries/requests that simulate real user needs **
3. We run each query against the system to obtain a ranked list of retrieved documents
4. We use the ranking and relevance judgements to calculate recall/precision pairs ***
5. Then we average recall and precision values across all queries

** The test collection should be large in order to capture user variation, diversity of subject matter, noise such as spelling mistakes, support claims of statistical significance in results, demonstrate that performance levels and differences hold as the document sizes grow, and have commercial credibility*.

*** since the database is so specialized, we could generate the queries from a pool of suggested queries provided by our relevance volunteers.*

**** Our volunteers provide the relevance judgement.*

Calculating further measures:

We need to calculate further measures to make sense of the results obtained for Recall and Precision:

- **Parametrized F measure (E measure)**

  The F-measure or F-score is calculated from the precision and recall.

  The standard balanced F_1 score is the harmonic mean of the precision and recall. The more generic parametrized F_β score applies additional weights, valuing more either precision or recall. The highest possible value of an F-score is 1.0, indicating perfect precision and recall, and the lowest possible value is 0, if either the precision or the recall is zero.

  We would use β > 1 in order to weight the recall more than the precision. Recall should be prioritized since it is vital not to miss any relevant existing patents when deciding whether to apply for a new patent. Precision can be somewhat sacrificed if it doesn't lead to unreasonably slow Query Response Times (see "efficiency").

$$F_\beta = \frac{(1+\beta^2)\cdot(\text{precision}\cdot\text{recall})}{(\beta^2\cdot\text{precision}+\text{recall})}$$

- Value of β controls the trade-off:
  - $\beta = 1$: equally weight precision and recall (E = F)
  - $\beta > 1$: weight recall more
  - $\beta < 1$: weight precision more

- **MAP (Mean Average Precision)**

  It is calculated over a test collection as the arithmetic mean of the average precision values. Each query is assumed to have the same weight. MAP assumes that the user is interested in finding many relevant documents for each query and is one of the measures most often used in IR of research papers. It values Recall above precision, which is ideal for our search system.

  Mean average precision for a set of queries is the mean of the average precision scores for each query:

$$\text{MAP} = \frac{\sum_{q=1}^{Q}\text{AveP(q)}}{Q}$$

  where Q is the number of queries.

Efficiency

Efficiency means how economically the system is achieving its objectives. In an information retrieval system efficiency can be measured by factors such as cost. The cost factors are to be calculated indirectly. They include factors such as response time, time taken by the system to provide an answer.

Measuring execution time of a query:

1. execute a query a sufficiently large number of times (for example, 50)
2. Reset the system between tests so we can get clean measures
3. Average the execution time measured

Effort

As previously mentioned, effort refers to the amount of work needed by the user to find and identify the relevant documents. This is a fairly subjective measure since it depends upon the viewpoint of the user. We will therefore need test users, ideally from all user target groups.

A good way to quantify the effort involved in using our search system is to use A/B testing (also known as split testing or bucket testing). In this process we perform a randomized experiment in which we show two variations of the patent search system to different groups of users at the same time (groups A and B, for variations A and B) and compare which version is considered to be easier to use by the users.

A/B testing would be appropriate for testing the new search system because we could compare it to the old search system and determine whether the new system is better than the old one. If it's not better (if it's worse or the same), there is no point in going through the expense of implementing it.

How to do A/B testing:

- The test users are divided into roughly equal size groups (of sizes n_1 and n_2) and assigned one of the two systems randomly.
- Null hypothesis: no difference between system A and B.
- Alternative hypothesis: there is a difference between the search systems A and B.

and the results are analysed with "two-sample hypothesis testing" which determines whether the results (and the difference in the results) is statistically significant or not.

Factors affecting our choice of test in our Two-sample hypothesis testing:

- Our data is categorical (e.g. can take on one of a limited, and usually fixed, number of possible values) coming from a discrete distribution over an ordinal scale:
    - We would use a 5-point Likert scale to measure each user's perception of the effort required (1 - difficult to use, 2 – somewhat difficult, 3 – neither difficult nor easy, 4 – somewhat easy, 5 – very easy to use).
- the hypothesis being tested is that there is a difference in the relevant population characteristics, which requires a two-sided test.
- The hypothesis being tested applies to a population parameter (we'll determine the central tendency using the mode or the median since we can't define a mean from ordinal data)

Software that we can use for testing:

We will want to use computer programs to do the data analysis for us. If it is available, I find that Microsoft's SPSS is quite straightforward to use and provides a wealth of tables and information (it is not the best option for graphics though). As Microsoft's SPSS is quite expensive, if it isn't already available then a better option would be to use R or Python.

Two-sample hypothesis testing:

- Two-sample test with Groups A (of size n_1) and B (of size n_2). The data are ordinal scores from 1 to 5 on the question "How easy is it to use this search system?".

  Wilcoxon rank sum test. Also called the Mann–Whitney U test. This would be a great option since it tests whether the ordinal values tend to be higher in one group than the other.

We do NOT want to use significance tests comparing the means - as a t-test would do. This would not be appropriate in our case, since the computation of means requires a cardinal scale which we do not have (we have an ordinal scale). This means that we cannot use the following (which would be good options for cardinal scales):

- **Welch t test**. If n_1 and n_2 are large enough (both above 20), we might be able to get a reliable answer using a Welch 2-sample t-test.
- **Permutation test**. This is often a good option. Under the null hypothesis that the two groups tend to give the same responses (e.g. there is no difference between the search systems), the argument is that the scores could be permuted between Groups A and B without effect. So if we choose some measure of difference such as the difference D = mean(X_1) – mean(X_2) between the two sample means (but we can't define the mean of an ordinal data set), we can use either simulation or combinatorics to get the null permutation distribution of D, and determine whether the observed value of D is consistent with the null distribution.

Robustness of the system

The robustness of the system can be tested through load testing and stress testing (also known as endurance testing).

Load Testing is to test the system behaviour under normal workload conditions, and it is just testing or simulating with the actual workload. Load testing does not break the system. Stress testing is to test the system behaviour under extreme conditions and is carried out till the system failure. Stress testing tries to break the system by testing with overwhelming data or resources. Appropriate error messages should appear when the system breaks and it should be possible to recover the system after it has broken down.

- To test how the system behaves in the extreme case of a large volume of searches/users we will need to do what is called "Exploratory Stress Testing"
- Stress testing's objective is to check the system under extreme conditions. It monitors system resources such as Memory, processor, network etc., and checks the ability of the system to recover back to normal status. It checks whether the system displays appropriate error messages while under stress.

The Stress Testing process can be done in 5 major steps:

1. Planning the Stress Test. Here you gather the system data, analyse the system, define the stress test goals.
2. Create Automation Scripts: In this phase, you create the Stress testing automation scripts, generate the test data for the stress scenarios.
3. Script Execution: In this stage, you run the Stress testing automation scripts and store the stress results.

4. Results Analysis: In this stage, you analyse the Stress Test results and identify bottlenecks.
5. Tweaking and Optimization: In this stage, you fine-tune the system, change configurations, optimize the code with goal meet the desired benchmark.

Lastly, you again run the entire cycle to determine that the tweaks have produced the desired results. For example, it's not unusual to have to 3 to 4 cycles of the Stress Testing process to achieve the performance goals.

A good tool to use for our required stress test would be Neo load. This is a popular tool available in the market to test the web and Mobile applications. This tool can simulate thousands of users in order to evaluate the application performance under load and analyse the response times. It also supports Cloud-integrated - performance, load and stress testing. It is easy to use, cost-effective and provides good scalability.

Part 4: (Expected) Results

Recall and Precision results:

Recall and precision are percentages and their value varies from 0 to 1. Since Recall is our key metric, we expect that Recall should either be 1 or very close to 1. Precision will have to be sacrificed and will therefore tend to be quite low.

The values for MAP and for F_β will depend on the Recall.

Recommendations for future improvements:

- Use **NDCG** (**N**ormalized **D**iscounted **C**umulative **G**ain) calculations in order to study the quality of the ranking of our query results since our current approach doesn't take ranking of results into account.

  People have no patience to scroll down long lists of results so the search results should be ordered by rank. Highly relevant documents should have better rank (=closer to the top). Highly relevant documents are more useful than marginally relevant documents, which are more useful than non-relevant documents – and their rank should reflect that.

  In our assessment of the patent search system we were more interested up to now in making sure that we had comprehensive retrieval but the order in which the search results are presented could prove time-saving for the users.

Efficiency (expected) results:

The two biggest components in measuring the response time of a query will be the query time and the server response time (since the performance of the database or the database server has a significant influence on the speed).

An average server response time of 200-350ms is considered fast, 400-700ms is average, and all the rest are "slow". Some servers may take longer to establish the connection, and others to transmit data. A query should take 20 to 500 ms (or sometimes more) depending on the system and the amount of data that is being sifted through.

We would expect the total Query Response time to vary between 220ms (=fast server + fast query) and 1200ms (=average server + complicated query). Longer times than that are an issue that should be addressed. "Average" response times are acceptable but "slow" times would be very frustrating for our users and should be avoided at all costs.

Recommendations for future improvements:

- If the query response time is "slow", it is essential to find the root cause and fix it. This is vital for user retention (beyond the first use, users will try to find alternative faster systems if our system is unreasonably slow)

Effort (expected) results:

At the end of our A/B testing and data analysis with two-sample hypothesis testing, we can obtain the following results:

1. The Null Hypothesis is accepted. There is no difference between the search systems.
2. The Null Hypothesis is rejected. There is a difference between the effort required by each of the two systems. The new system is found to be easier to use.
3. The Null Hypothesis is rejected. There is a difference between the effort required by each of the two systems. The old system is found to be easier to use.

Recommendations:

- Results 1. and 2. are both acceptable outcomes. The system should be at least as easy to use as the old system (outcome 1.) or, in the optimal scenario, require less effort to use (outcome 2.).
- Result 3. is not an acceptable outcome. If the new search system requires more effort than the old search system, users will prefer to use the old system. The features that make the search system require more effort should be identified and addressed. For example:
    - the "difficult" features could be altered to make them more intuitive
    - additional features could be added to the system's UI to guide the user, such as "help" pop-ups when scrolling the mouse over certain areas

Robustness of the system (expected) results:

After extensive Stress testing, our system should be able to withstand large volumes of queries and simultaneous users.

Recommendations for future improvements:

- Keep an eye on the server response time creeping up when there is a lot of traffic. This can cause the overall query response times to increase and become "slow". If there is a limit after which the response time becomes "very slow", it might be better to limit the amount of active queries/users. It is better if some users get a message of "High traffic; please try again in a few minutes" and the other users have good query response times than all user have slow query response times. This way the majority of the users will be satisfied.

Other Recommendations for future improvements:

- (***Important and suggested as a requirement***) Add a feature for suggesting new search terms based upon the user's current query. This would improve recall and increase the likelihood of finding all relevant documents. **This was also one of the requirements for some of the users that were identified in our interviews**.

- **Features that might be made available with a paid subscription or a pay-per-case:**
    - Print/save the documents associated with the selected patents.
    - Ability to save search results for future use. This would add to the "user-friendly" experience and make the search system easier to use.

Some Sources:

- [1] World Intellectual Property organization Patent Information FAQ: https://www.wipo.int/patents/en/faq_patents.html
- Lecture notes from the course "Information Retrieval"
- Search Trademark Database: https://www.uspto.gov/trademarks/search
- Stress testing: https://www.guru99.com/stress-testing-tutorial.html#:~:text=Stress%20Testing%20is%20a%20type,t%20crash%20under%20crunch%20situations
- Evaluation of Information Retrieval Systems (purpose and criteria):

  https://upload.wikimedia.org/wikipedia/commons/1/17/Evaluation_of_information_retrieval_system_purpose_and_retrieval.pdf
- Hot to use interviews to gather requirements: https://reqtest.com/requirements-blog/how-to-use-interviews-to-gather-requirements/
- Two-sample tests for ordinal data: https://math.stackexchange.com/questions/2121411/two-sample-test-for-ordinal-data
- Evaluation measures in Information Retrieval: https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)