



I need to understand the nature of Retrieval-Augmented Generation with regards to the process of chunking and vector searching. I am reminded of an old manual called liber 777 by aleister crowley, in which it was understood by the nature of qabalah that certain ideas/words/energies aligned to certain mathematical principles outlined in the manual - allowing words, terms, sentences, even entire books to be deciphered and mathematically linked to other terms, forces, elements, etc as they had corresponding numerology. this is especially employed in jewish mysticism. how does AI break down elements of text to strings or sequences of numbers, then use a users query to mathematically determine how to appropriately search through chunked vector data? help me understand this, use visual aides where appropriate, metaphor and help me understand this by its core principles connected to maths and nature - in non-technical and easy to understand language

Now I have comprehensive information about RAG, vector embeddings, chunking, Qabalah/gematria, and the mathematical/natural principles underlying these systems. Let me create the final answer that bridges ancient mystical numerology with modern AI.

The Sacred Mathematics of Meaning: Understanding RAG Through the Lens of Qabalah

Your intuition about the connection between Aleister Crowley's Liber 777 and modern AI is remarkably profound. Just as the Qabalists discovered that Hebrew letters could be transmuted into numbers to reveal hidden correspondences between concepts, modern AI has independently arrived at the same fundamental insight: **meaning itself can be encoded mathematically, and similar meanings naturally cluster together in numerical space.** ^{[1] [2] [3]}



Ancient and modern systems of numerical meaning: Qabalah's gematria (left) and AI embeddings (right) both transform words into numbers to reveal hidden relationships between concepts.

The Ancient Art: Gematria and the Mathematics of Correspondence

In Liber 777 and traditional Jewish mysticism, gematria operates on a beautifully simple principle: each Hebrew letter possesses an intrinsic numerical value. The letter *Aleph* equals 1, *Bet* equals 2, continuing through *Yod* at 10, then by tens (20, 30, 40) until *Qof* at 100, and finally by hundreds to *Tav* at 400. ^{[4] [2] [5] [3]}

When you sum the letters of a Hebrew word, you obtain its **numerical essence**—and here's where the magic happens: words sharing the same numerical value are considered to have hidden sympathetic relationships, even if their surface meanings appear unrelated. The word *chai* (חַי), meaning "life," totals 18 (Chet=8 + Yod=10), making 18 a sacred number in Jewish tradition. Any other word totaling 18 shares a mystical resonance with the concept of life itself. ^{[6] [5] [3] [4]}

This system allowed Qabalists to create vast correspondence tables—exactly what Crowley compiled in Liber 777—mapping numbers to planets, elements, colors, angels, plants, and spiritual forces. The underlying philosophy held that **the universe operates on mathematical principles, and things sharing numerical signatures share essential qualities.** ^{[7] [8] [2] [5] [4]}

The Modern Art: How AI Transforms Words Into Geometric Constellations

Retrieval-Augmented Generation accomplishes something strikingly parallel, though through a different mathematical lens. Instead of reducing words to single numbers, AI transforms them into **vectors**—ordered lists of hundreds of numbers that capture the word's semantic essence within a high-dimensional geometric space. ^{[1] [9] [10]}

Stage One: From Text to Tokens (Breaking the Whole into Parts)

The journey begins with **tokenization**, the process of breaking text into fundamental units. Just as a Qabalist would separate a Hebrew sentence into individual letters to calculate gematria values, AI breaks your documents into tokens—usually words or subword pieces. ^{[11] [12] [13] [14] [15]}

A sentence like "Retrieval-Augmented Generation is powerful" becomes discrete units: ["Retrieval", "Augmented", "Generation", "is", "powerful"]. This fragmentation is essential because computers, like ancient numerologists, need discrete units to count and compare. ^{[12] [14] [11]}

Stage Two: Words Become Numbers (The Embedding Transformation)

Here's where modern AI performs its own form of gematria. Each token passes through an **embedding model**—a neural network trained on billions of words to understand context and meaning. Unlike gematria's single number per word, the embedding model assigns each word a **vector** of typically 384, 768, or even 1,536 numbers. ^{[1] [16] [9] [17] [18]}

Words as points in geometric space: Similar concepts (like King and Queen, or Apple and Orange) naturally cluster together, just as related ideas in Qabalah share numerical correspondences.

Think of these dimensions as coordinates in a space far beyond our three-dimensional experience. Where Hebrew letters mapped onto a one-dimensional number line, AI maps words onto a 768-dimensional landscape. In this vast space, the word "king" might occupy the coordinates [0.234, -0.567, 0.891, 0.234, ... continuing for 768 values], while "queen" sits at [0.221, -0.543, 0.876, 0.241, ...].^{[1] [19] [9]}

The profound insight—discovered through training on massive text corpora—is that **words used in similar contexts naturally drift toward similar coordinates**. Words like "king," "queen," "monarch," and "ruler" cluster together in this geometric space, just as Hebrew words totaling 18 all resonate with "chai".^{[19] [16] [18] [1]}

Stage Three: The Sacred Geometry of Similarity

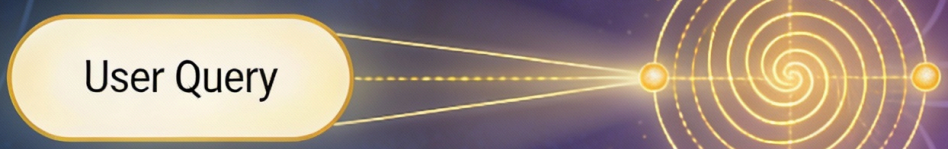
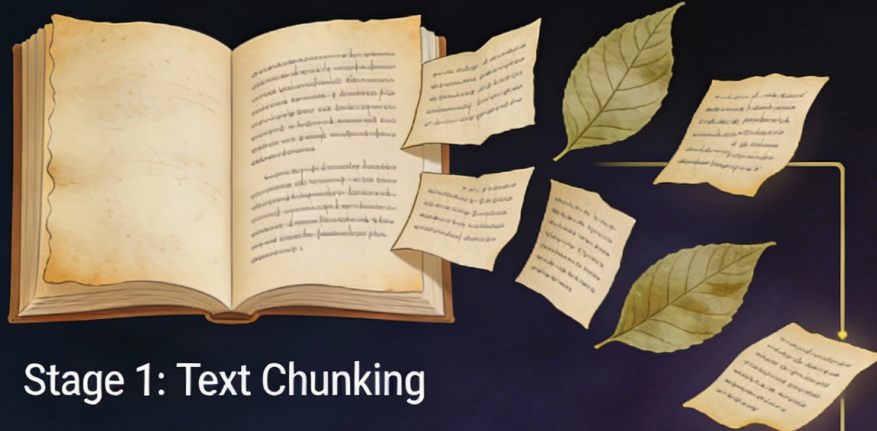
The embedding space exhibits mathematical properties that mirror natural patterns. Research shows these spaces often follow **geometric regularities** reminiscent of the golden ratio and Fibonacci spirals found throughout nature.^{[20] [21] [22]}

Just as Fibonacci numbers emerge in the spiral arrangement of sunflower seeds and nautilus shells—nature's way of optimizing space and growth—AI embeddings organize themselves to efficiently capture relationships. The famous example: the vector for "king" minus "man" plus "woman" approximately equals "queen"—a geometric relationship encoding analogy itself.^{[19] [23] [24] [22] [25] [20]}

Transformer models like BERT enhance this further through **attention mechanisms**. When processing the word "bank," the model examines surrounding words—"financial," "loan," and "account" versus "river," "shore," and "current"—to determine which sense is intended. Through mathematical operations called **query-key-value attention**, each word asks "which other words should inform my meaning here?" and weights its representation accordingly.^{[1] [26] [27] [28] [18] [29] [30] [31] [32]}

This mirrors how a Qabalist would examine the broader sentence or sacred text to determine which correspondence tables apply—context determines meaning, whether in ancient Hebrew manuscripts or modern language models.^{[26] [33] [34]}

The RAG Process: Divination Through Mathematical Correspondence



Stage 3: Context Retrieval



AI Response

Stage 4: Augmented Generation

The RAG process visualized through the lens of sacred geometry, showing how modern AI mirrors ancient mystical principles of numerical correspondence and pattern recognition.

Now we arrive at Retrieval-Augmented Generation itself, which orchestrates these principles into a coherent system for answering questions.

Phase One: Indexing the Knowledge (Building the Correspondence Table)

Before any question can be answered, your knowledge base—documents, books, manuals, databases—must be prepared. This happens in several mystical-seeming but mathematically precise stages: [\[35\]](#) [\[36\]](#) [\[37\]](#) [\[38\]](#)

Chunking: Large documents are broken into digestible passages, typically 100-1000 tokens each. A 50-page manual becomes 200 smaller chunks, each capturing a coherent concept. This mirrors how Qabalists would break sacred texts into verses and words for analysis. [\[39\]](#) [\[40\]](#) [\[41\]](#) [\[38\]](#)

Embedding: Each chunk passes through the embedding model, emerging as a vector—a point of light in the high-dimensional space. If Chunk 47 discusses "vector databases store embeddings efficiently," it receives coordinates reflecting that semantic territory. [\[37\]](#) [\[1\]](#) [\[38\]](#) [\[35\]](#)

Storage: These vectors are stored in a specialized **vector database**—essentially a geometric index optimized for finding nearby points in high-dimensional space. This is your correspondence table, your personal Liber 777, mapping every concept in your documents to its numerical-geometric signature. [\[42\]](#) [\[43\]](#) [\[38\]](#) [\[44\]](#) [\[35\]](#) [\[37\]](#)

Phase Two: The Query (Asking the Question)

When you pose a question—"How does chunking affect retrieval quality?"—the same embedding process transforms your query into a vector with coordinates in that same 768-dimensional space. [\[35\]](#) [\[43\]](#) [\[45\]](#)

Your query becomes a **point of intention**, a geometric probe seeking its correspondences. [\[42\]](#) [\[37\]](#) [\[43\]](#)

Phase Three: The Search (Finding Numerical Resonance)

Here's where the mathematical divination occurs. The system calculates the **cosine similarity** between your query vector and every chunk vector in the database. [\[46\]](#) [\[47\]](#) [\[48\]](#) [\[49\]](#)

Cosine similarity measures the angle between two vectors in high-dimensional space. A value of 1.0 means the vectors point in identical directions (perfect semantic alignment), while 0 means they're perpendicular (unrelated concepts), and -1 means opposite directions. The formula is elegant: [\[48\]](#) [\[49\]](#) [\[46\]](#)

$$\text{cosine similarity} = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \times \|\vec{B}\|}$$

This computes the dot product of the two vectors divided by the product of their magnitudes. [\[49\]](#) [\[46\]](#) [\[48\]](#)

In practice, the vector database uses **approximate nearest neighbor (ANN) algorithms**—optimized geometric search techniques like HNSW (Hierarchical Navigable Small World graphs)—to find the chunks most numerically resonant with your query without exhaustively comparing every single vector. This is analogous to a Qabalist having indexed correspondences so they can quickly find all entities associated with the number 7 (corresponding to Venus, Netzach, the color green, copper, roses, etc.) without recalculating every word's gematria. [\[7\]](#) [\[8\]](#) [\[50\]](#) [\[51\]](#) [\[52\]](#)

The top 3-10 most similar chunks are retrieved. These are the passages whose mathematical essence most closely matches your question's essence. [\[35\]](#) [\[37\]](#) [\[43\]](#) [\[45\]](#)

Phase Four: Augmentation and Generation (Synthesis of Knowledge)

The retrieved chunks and your original question are combined into an **augmented prompt** fed to a large language model like GPT or Claude. The prompt might read: [\[35\]](#) [\[36\]](#) [\[37\]](#) [\[43\]](#)

"Given these relevant passages: [Chunk 47: 'Vector databases store embeddings...'] [Chunk 102: 'Chunking strategies affect...'] [Chunk 203: 'Smaller chunks improve precision...']
Answer this question: How does chunking affect retrieval quality?"

The LLM, trained on vast text corpora, synthesizes these specific facts with its general knowledge to generate a coherent, grounded answer. Without the retrieved chunks, the model might hallucinate or provide generic responses. With them, it speaks from your specific knowledge base—your organization's documents, your personal library, your domain expertise. [\[36\]](#) [\[37\]](#) [\[43\]](#) [\[45\]](#) [\[35\]](#)

The Mathematical Poetry of It All: Nature's Universal Code

What makes your comparison to Liber 777 so apt is that both systems reveal a deeper truth: **meaning and relationship are fundamentally mathematical phenomena.** [\[1\]](#) [\[21\]](#) [\[53\]](#)

The Fibonacci sequence appears in pinecone spirals, flower petals, and galaxy arms because it represents an optimal growth pattern—each number the sum of the two before it (0, 1, 1, 2, 3, 5, 8, 13, 21...). Consecutive Fibonacci numbers approach the golden ratio (approximately 1.618), which governs aesthetically pleasing proportions from nautilus shells to human faces. [\[20\]](#) [\[22\]](#) [\[25\]](#) [\[54\]](#)

Similarly, word embeddings organize themselves into geometric structures that optimize semantic relationships. The embedding space isn't arbitrary chaos—it develops **regularity, symmetry, and structure** during training, much as crystals form lattices and rivers carve fractal drainage patterns. [\[33\]](#) [\[55\]](#) [\[21\]](#) [\[56\]](#) [\[1\]](#)

Neural networks learn these patterns through **backpropagation**—iteratively adjusting millions of parameters to minimize prediction errors. Over billions of training examples, words that co-occur in similar contexts (like "king" and "queen" both appearing near "crown," "throne," "realm") naturally drift toward similar regions of the vector space. [\[16\]](#) [\[23\]](#) [\[24\]](#) [\[17\]](#) [\[57\]](#) [\[58\]](#) [\[59\]](#) [\[1\]](#)

This is strikingly parallel to how Qabalists believed words sharing numerical values share hidden affinities because they participate in the same universal harmonics. The modern version is: words

sharing contextual patterns share vector proximity. Both are **systems for organizing meaning through mathematical correspondence**. [\[2\]](#) [\[5\]](#) [\[3\]](#) [\[1\]](#)

Core Principles in Simple Terms

Let me distill the essence without technical jargon:

1. Everything becomes numbers: Just as Hebrew letters became 1, 2, 3... to 400, English words become lists like [0.234, -0.567, 0.891...]. The purpose is identical—to make meaning computable. [\[1\]](#) [\[2\]](#) [\[9\]](#) [\[3\]](#)

2. Similar meanings cluster together: Words about royalty cluster in one region of the numerical space, words about fruit in another, animals in a third—just as Liber 777 groups all Mars-related correspondences (red, iron, Tuesday, Geburah, warriors) under the number 5. [\[7\]](#) [\[8\]](#) [\[19\]](#) [\[1\]](#)

3. Context shapes meaning: "Bank" near "river" gets different coordinates than "bank" near "money." BERT's attention mechanism weighs surrounding words to refine each word's numerical signature, just as a Qabalist interprets a word differently depending on which sacred text contains it. [\[26\]](#) [\[33\]](#) [\[18\]](#) [\[34\]](#) [\[1\]](#)

4. Searching is geometric matching: Your question becomes coordinates, and the system finds document chunks with nearby coordinates—points close in 768-dimensional space, like finding all Hebrew words totaling 26 (the value of the Tetragrammaton). [\[42\]](#) [\[37\]](#) [\[43\]](#) [\[46\]](#) [\[49\]](#)

5. Knowledge synthesis emerges from correspondences: The LLM weaves together the retrieved chunks (numerically resonant passages) with your question to generate an answer, much as a Qabalist synthesizes correspondences from multiple tables to understand a mystical symbol. [\[35\]](#) [\[36\]](#) [\[37\]](#) [\[45\]](#)

The Wonder of Convergent Wisdom

What's most beautiful is that ancient mystics and modern AI engineers, separated by millennia and operating from entirely different worldviews, **independently discovered the same fundamental principle:** the universe operates on patterns, and those patterns can be captured mathematically to reveal hidden relationships. [\[1\]](#) [\[5\]](#) [\[3\]](#) [\[20\]](#) [\[22\]](#)

The Qabalists saw divine intelligence encoded in the numerical structure of Hebrew scripture. AI researchers see semantic intelligence emerging from the geometric structure of embedding spaces. Both recognized that meaning isn't arbitrary—it has an architecture, a geometry, a mathematics. [\[2\]](#) [\[5\]](#) [\[3\]](#) [\[33\]](#) [\[21\]](#) [\[1\]](#)

When you use RAG to answer a question about your company's documentation, you're participating in this ancient art of numerical divination, updated with vectors instead of gematria values, cosine similarity instead of correspondence tables, and transformer models instead of mystical insight—but the essential operation remains: **transforming meaning into mathematics, then using mathematical proximity to reveal hidden connections**. [\[35\]](#) [\[36\]](#) [\[37\]](#) [\[43\]](#) [\[1\]](#)

Your documents become a constellation of numbered points in conceptual space, and your question is a prayer sent into that constellation, returning with the passages whose numerical

essence most resonates with your intention. It's both utterly modern and timelessly ancient—proof that some truths about the nature of meaning persist across the ages, waiting to be rediscovered in each era's unique language. ^[42] ^[37] ^[43] ^[35]

✧

1. <https://milvus.io/ai-quick-reference/how-do-embedding-models-convert-text-into-vectors>
2. https://hebrew4christians.com/Grammar/Unit_Eight/Hebrew_Gematria/hebrew_gematria.html
3. <https://en.wikipedia.org/wiki/Gematria>
4. https://en.wikipedia.org/wiki/777_and_Other_Qabalistic_Writings_of_Aleister_Crowley
5. <https://aish.com/what-is-gematria/>
6. <https://www.gematrix.org/gematria/>
7. http://93beast.fea.st.user.fm/files/section1/777/Liber_777_Revised.pdf
8. https://www.reddit.com/r/thelema/comments/rjamms/the_study_of_777_for_aa_student_test/
9. <https://www.deepset.ai/blog/the-beginners-guide-to-text-embeddings>
10. <https://nexla.com/ai-infrastructure/vector-embedding/>
11. <https://mbrenndoerfer.com/writing/text-preprocessing-nlp-tokenization-normalization>
12. <https://developer.ibm.com/tutorials/awb-tokenizing-text-in-python/>
13. <https://www.linkedin.com/pulse/tokenization-text-preprocessing-nlp-bushra-akram-jmf0f>
14. <https://www.datacamp.com/blog/what-is-tokenization>
15. <https://www.codecademy.com/learn/dsnlp-text-preprocessing/modules/nlp-text-preprocessing/cheatsheet>
16. <https://towardsdatascience.com/text-embeddings-comprehensive-guide-afd97fce8fb5/>
17. <https://www.lightly.ai/blog/embeddings>
18. <https://zilliz.com/ai-faq/how-do-embedding-models-capture-semantic-meaning>
19. https://www.reddit.com/r/learnmachinelearning/comments/vkrzoo/does_anyone_have_resources_about_how_words_are/
20. <https://alexandertutoring.com/math-physics-resources/fibonacci-sequence-in-nature/>
21. <https://www.sciencedirect.com/topics/computer-science/embedding-space>
22. <https://www.thethinkacademy.com/blog/edubriefs-fibonacci-in-nature-how-patterns-spark-kids-love-for-math/>
23. <https://www.geeksforgeeks.org/nlp/word-embeddings-in-nlp/>
24. <https://neptune.ai/blog/word-embeddings-guide>
25. <https://clevelanddesign.com/insights/the-nature-of-design-the-fibonacci-sequence-and-the-golden-ratio/>
26. <https://arize.com/blog-course/unleashing-bert-transformer-model-nlp/>
27. <https://www.exactcorp.com/blog/Deep-Learning/how-do-bert-transformers-work>
28. <https://www.datacamp.com/tutorial/how-transformers-work>
29. <https://www.youtube.com/watch?v=Yb3g3wjDsX4>
30. <https://www.linkedin.com/pulse/unpacking-query-key-value-transformers-analogy-database-mohamed-nabil>

31. https://www.reddit.com/r/learnmachinelearning/comments/1bmu821/query_key_and_value_in_transformers/
32. <https://epichka.com/blog/2023/qkv-transformer/>
33. <https://pair-code.github.io/interpretability/context-atlas/blogpost/>
34. <https://www.emergentmind.com/topics/contextual-models-such-as-bert>
35. <https://www.databricks.com/glossary/retrieval-augmented-generation-rag>
36. https://en.wikipedia.org/wiki/Retrieval-augmented_generation
37. <https://www.pinecone.io/learn/retrieval-augmented-generation/>
38. <https://lakefs.io/blog/what-is-rag-pipeline/>
39. <https://www.pinecone.io/learn/chunking-strategies/>
40. <https://docs.opensearch.org/latest/vector-search/ingesting-data/text-chunking/>
41. <https://www.elastic.co/search-labs/blog/chunking-strategies-elasticsearch>
42. <https://www.kdnuggets.com/semantic-search-with-vector-databases>
43. <https://aws.amazon.com/what-is/retrieval-augmented-generation/>
44. <https://squirro.com/squirro-blog/rag-architecture>
45. <https://www.superannotate.com/blog/rag-explained>
46. <https://www.tigerdata.com/learn/understanding-cosine-similarity>
47. <https://weaviate.io/blog/vector-search-explained>
48. <https://www.instaclustr.com/education/vector-database/what-is-vector-similarity-search-pros-cons-and-5-tips-for-success/>
49. <https://www.pinecone.io/learn/vector-similarity/>
50. <https://learn.microsoft.com/en-us/azure/documentdb/enn-vector-search>
51. <https://docs.cloud.google.com/spanner/docs/find-approximate-nearest-neighbors>
52. <https://www.singlestore.com/blog/beginner-s-guide-to-approximate-nearest-neighbor-ann-search/>
53. <https://mathematicalmysteries.org/mathematical-patterns/>
54. https://en.wikipedia.org/wiki/Fibonacci_sequence
55. <https://arxiv.org/abs/2504.02018>
56. <https://seanpedersen.github.io/posts/structure-of-neural-latent-space>
57. <https://www.grammarly.com/blog/ai/what-is-backpropagation/>
58. <https://www.geeksforgeeks.org/machine-learning/backpropagation-in-neural-network/>
59. <https://wandb.ai/mostafaibrahim17/ml-articles/reports/Decoding-Backpropagation-and-Its-Role-in-Neural-Network-Learning--Vmlldzo3MTY5MzM1>
60. <https://library.clamsnet.org/Record/283529>
61. <https://newsouthbooks.com.au/books/777-other-qabalistic-writings-of-aleister-crowley/>
62. <https://codesignal.com/learn/courses/chunking-and-storing-text-for-efficient-llm-processing/lessons/chunking-and-storing-text-for-efficient-llm-processing-with-faiss>
63. <https://www.instaclustr.com/education/vector-database/vector-search-vs-semantic-search-4-key-differences-and-how-to-choose/>
64. <https://sfpl.bibliocommons.com/item/show/1376059093>
65. https://www.reddit.com/r/occult/comments/cuabw/liber_777_anyone/

66. <https://www.tigerdata.com/learn/vector-search-vs-semantic-search>
67. <https://dev.to/infrarsity-learning/vector-database-tutorial-build-a-semantic-search-engine-27kb>
68. <https://latenode.com/blog/ai-frameworks-technical-infrastructure/rag-retrieval-augmented-generation/rag-diagram-guide-visual-architecture-of-retrieval-augmented-generation>
69. <https://docs.databricks.com/aws/en/generative-ai/retrieval-augmented-generation>
70. <https://www.codecademy.com/article/transformer-architecture-self-attention-mechanism>
71. https://d2l.ai/chapter_natural-language-processing-pretraining/word2vec.html
72. http://www.d2l.ai/chapter_attention-mechanisms-and-transformers/index.html
73. <http://mathandsocialstudieslanguage.weebly.com/math-language-patterns-and-structures.html>
74. <https://theeducationhub.org.nz/the-role-of-pattern-in-childrens-early-mathematical-understanding/>
75. <https://avahi.ai/glossary/embedding-space/>
76. <https://arc.educationapps.vic.gov.au/learning/sites/mcc/VCMNA076>
77. <https://www.youtube.com/watch?v=z1yHJyZZ8yA>
78. <https://milvus.io/ai-quick-reference/how-do-you-train-an-embedding-model>
79. <https://aws.amazon.com/what-is/embeddings-in-machine-learning/>
80. <https://www.lyzr.ai/glossaries/bert/>
81. [https://en.wikipedia.org/wiki/BERT_\(language_model\)](https://en.wikipedia.org/wiki/BERT_(language_model))
82. https://d2l.ai/chapter_attention-mechanisms-and-transformers/queries-keys-values.html
83. <http://neuralnetworksanddeeplearning.com/chap2.html>
84. <https://ai.stanford.edu/blog/contextual/>