```asciidoc
= Vectors & Semantic Search
:order: 1
:type: lesson

In the last module, you learned about the importance of grounding to
improve LLM accuracy and the concept of **Retrieval Augmented Generation**
(RAG).
RAG involves providing additional information to help the LLM form a
response.

One of the challenges of RAG is understanding what the user is asking for
and surfacing the correct information to pass to the LLM.

In this lesson, you will learn about semantic search and how vector
indexes can help you implement it in Neo4j.

[.video]
video::2Z81g1S54i4[youtube,width=560,height=315]

[.transcript]
== Vectors and Semantic Search

Semantic search aims to understand search phrases' intent and contextual
meaning, rather than focusing on individual keywords.

Traditional keyword search often depends on exact-match keywords or
proximity-based algorithms that find similar words.

For example, if you input "apple" in a traditional search, you might
predominantly get results about the fruit.

However, in a semantic search, the engine tries to gauge the context: Are
you searching about the fruit, the tech company, or something else?

The results are tailored based on the term and the perceived intent.

=== Vectors and embeddings

In natural language processing (NLP) and machine learning, numerical
representations (known as **vectors**) represent words and phrases.

Each dimension in a vector can represent a particular semantic aspect of
the word or phrase.
When multiple dimensions are combined, they can convey the overall meaning
of the word or phrase.

A vector will not directly encode tangible attributes like color, taste,
or shape.
Instead, the model will generate a list of numerical values that closely
align the word with related words such as health, nutrition, and wellness.

When applied in a search context, the vector for "apple" can be compared
to the vectors for other words or phrases to determine the most relevant
```

```
results.

You can create vectors in various ways, but one of the most common methods
is to use a **large language model**. These vectors are known as
**embeddings**.
With advanced models, these embeddings also contain contextual
information.

For example, the embeddings for the word "apple" are `0.0077788467,
-0.02306925, -0.007360777, -0.027743412, -0.0045747845, 0.01289164,
-0.021863015, -0.008587573, 0.01892967, -0.029854324, -0.0027962727,
0.020108491, -0.004530236, 0.009129008,` ... and so on.

[%collapsible]
.Reveal the completed embeddings for the word "apple"!
====
[source]
----
include::includes/apple.txt[]
----
====

The vector for a word can change based on its surrounding context. For
instance, the word _bank_ will have a different vector in _river bank_
than in _savings bank_.

Semantic search systems can use these contextual embeddings to understand
user intent.

[NOTE]
.Creating Vector Embeddings
====
LLM providers typically expose API endpoints that convert a _chunk_ of
text into a vector embedding.
Depending on the provider, the shape and size of the vector may differ.

For example, OpenAI's `text-embedding-ada-002` embedding model converts
text into a vector of 1,536 dimensions.
====

You can use the _distance_ or _angle_ between vectors to gauge the
semantic similarity between words or phrases.

image::images/vector-distance.svg[A 3 dimensional chart illustrating the
distance between vectors. The vectors are for the words "apple" and
"fruit"]

Words with similar meanings or contexts will have vectors that are close
together, while unrelated words will be farther apart.

This principle is employed in semantic search to find contextually
relevant results for a user's query.
```

A semantic search involves the following steps:

. The user submits a query.
. The system creates a vector representation (embedding) of the query.
. The system compares the query vector to the vectors of the indexed data.
. The results are scored based on their similarity.
. The system returns the most relevant results to the user.

image::images/semantic-vector-search.svg[A diagram show the steps of a semantic search.]

Vectors can represent more than just words. They can also represent entire documents, images, audio, or other data types. They are instrumental in the operation of many other machine-learning tasks.

== Vectors and Neo4j

Vectors are the backbone of semantic search. They enable systems to understand and represent the complex, multi-dimensional nature of language, context, and meaning.

Neo4j supports link:https://neo4j.com/docs/cypher-manual/current/indexes-for-vector-search/[vector indexes^] and querying, allowing you to search for nodes based on their vector representations.

In the next lesson, you will learn how to use vectors to implement semantic search in Neo4j.

[.checklist]

== Check Your Understanding

include::questions/1-semantic-vs-traditional.adoc[leveloffset=+1]
include::questions/2-vector-role.adoc[leveloffset=+1]

[.summary]
== Lesson Summary

In this lesson, you learned how semantic search differs from traditional keyword search. Vectors, representing data numerically, facilitate this advanced search mechanism and are integral to many machine learning algorithms like LLMs.

In the next lesson, you will learn how to use vector indexes in Neo4j and implement semantic search.