**EXAMPLE WORKFLOW: INTUITIVE UNDERSTANDING OF EMBEDDING**

Let’s imagine we’re training a **text classifier** that detects **sentiment** (positive or negative), and we have the following three sentences:

1. "I love this movie" → **Positive**
2. "I hate this movie" → **Negative**
3. "This film is awesome" → **Positive**

**Step 1: Word Representation (Initial State)**

Initially, we don't know anything about the words, and that target dimension is 4. So we **randomly assign** each word a small vector (say 4 dimensions):

| **Word** | **Vector (random)** |
| --- | --- |
| I | [0.1, -0.2, 0.05, 0.07] |
| love | [0.3, 0.6, -0.1, -0.4] |
| hate | [-0.4, 0.1, 0.7, -0.2] |
| movie | [0.2, -0.3, 0.1, 0.5] |
| awesome | [0.5, 0.3, 0.1, 0.2] |
| film | [0.15, -0.2, 0.05, 0.4] |

These are the **embeddings** stored in the **embedding layer's weight matrix**.

**Step 2: Feed to Neural Network**

Now we pass sentence 1: "I love this movie" → [I, love, this, movie]

* These are converted into their **vectors** using the current **embedding layer**.
* The combined vector (through average, concatenation, or some pooling) is passed into a neural network.
* The network **predicts sentiment**, say 0.6 (slightly positive), but we know the **true label is 1.0 (positive)**.

**Step 3: Backpropagation Updates the Embedding**

Because the prediction is off (0.6 vs 1.0), the network computes **loss**, and backpropagation begins.

* Gradients flow **back from the output**, into the **hidden layers**, and finally into the **embedding layer**.
* Only the embeddings of words **present in the sentence** get updated (e.g., I, love, this, movie).

Their vectors are adjusted slightly so that, next time, the sentence pushes the prediction **closer to the true label**.

**Step 4: Repeating for Other Sentences**

Now we pass sentence 2: "I hate this movie" (label = 0)

* Again, the network makes a prediction, compares with the true label (negative), and **adjusts the vectors** of the words like hate, movie, this.

Over time, embeddings of **words in positive contexts (love, awesome)** become similar, and **words in negative contexts (hate)** drift apart.

**Step 5: Learned Semantic Structure**

Eventually, the model learns:

* love and awesome have **similar embeddings**
* hate is far from love
* movie and film become **similar** due to shared contexts

So when new sentences like "That film is lovely" appear, even if the model hasn’t seen “lovely,” it may still classify it correctly based on the learned **similarity in embedding space**.

To keep it simple, the **embedding layer is just a lookup table** that is updated **via backpropagation** like other weights in the neural network, based on how well the current embeddings are able to produce an output in alignment with target, and what changes must be done to make its output prediction more in alignment with target value**.** In this process, it **compresses and encodes relationships** between words using the **training task’s feedback**.It is not manually created or designed — it **emerges from data**.