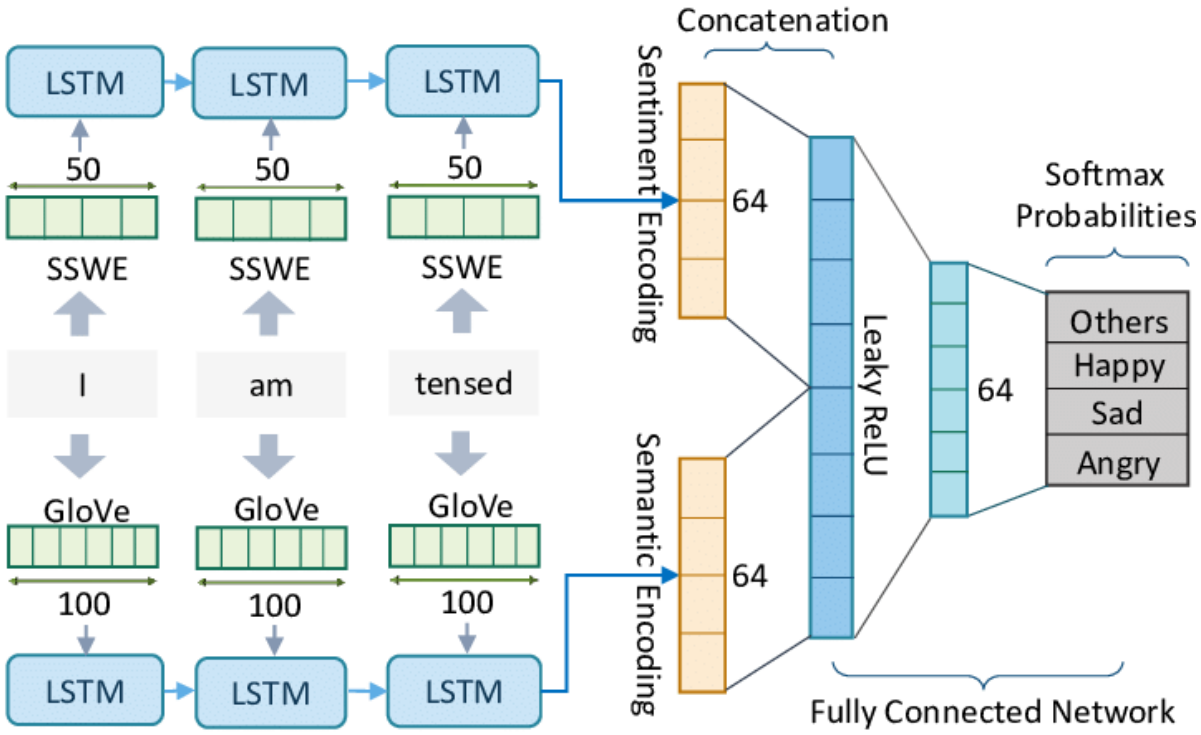
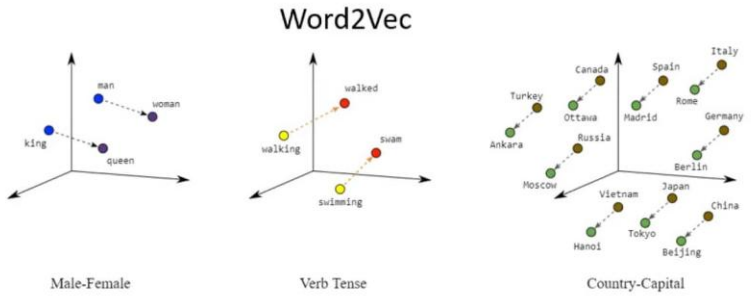
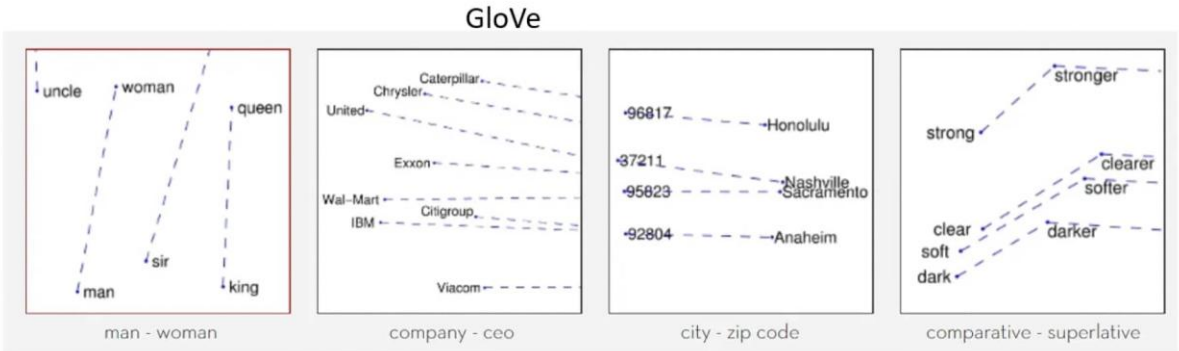
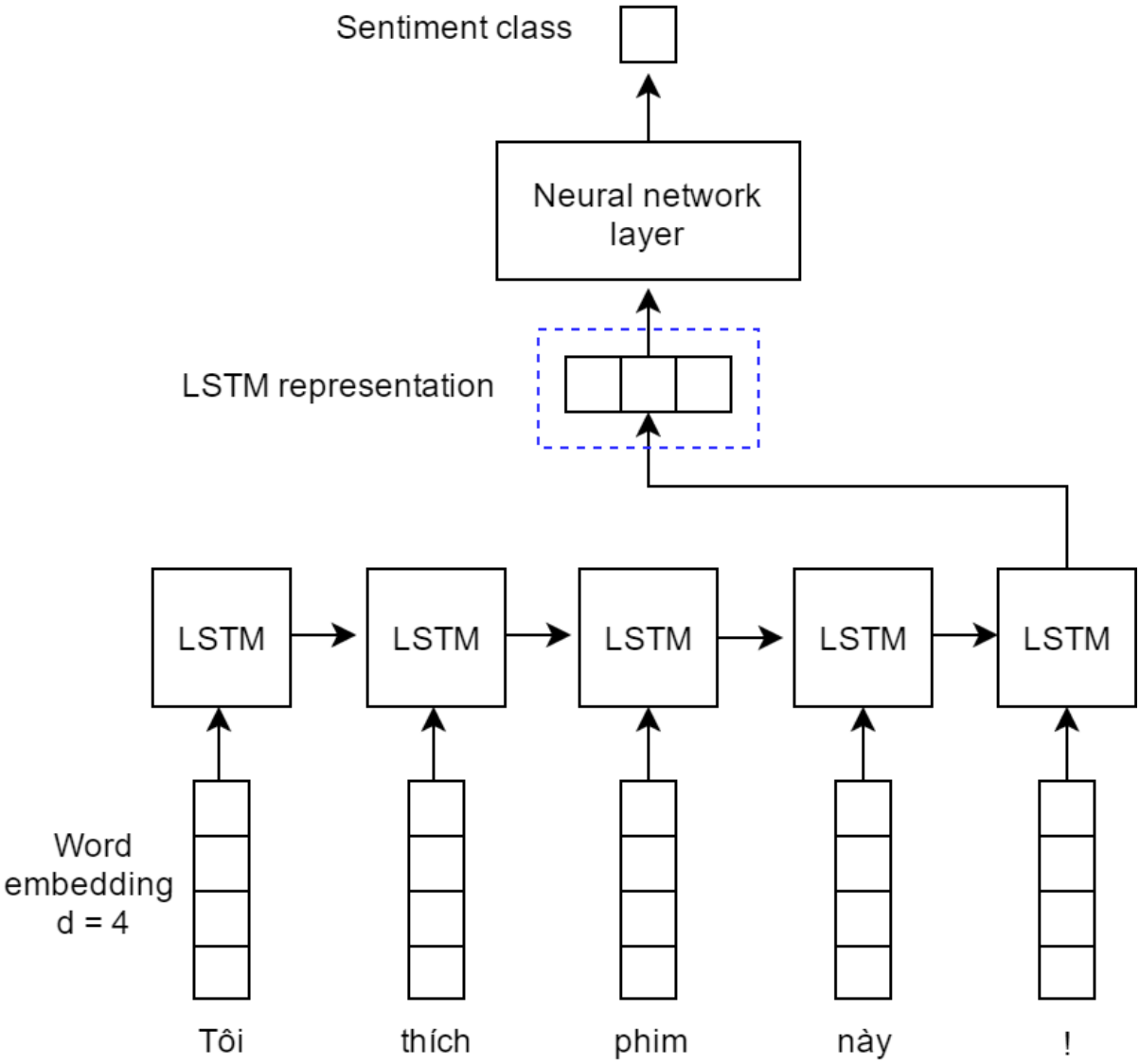
How Sentiment Analysis Works Using Numerical Values (LSTM)









We'll break this into **5 logical stages**:

1️⃣ **Why Text MUST Be Converted to Numbers**

👉 **Neural networks cannot understand text directly**
They only work with **numbers (vectors)**.

So this text:

"Power outage in my area since 10 hours"

❌ cannot be given directly to LSTM
✅ must be converted into **numerical form**

2️⃣ Step 1: Text → Tokens (Words)

The sentence is first **tokenized**:

["power", "outage", "in", "my", "area", "since", "10", "hours"]

Each word is now a **unit in a sequence**.

3️⃣ Step 2: Tokens → Numerical Values (Word Indexing)

We create a **vocabulary dictionary** during training:

Word	Index
power	15
outage	87
area	42
hours	109
emergency	301

Now the sentence becomes:

[15, 87, 9, 21, 42, 55, 7, 109]

📌 This is the **first numerical representation**
But these numbers **do NOT yet carry meaning**

4️⃣ Step 3: Embedding Layer (MOST IMPORTANT)

❓ What is an Embedding?

An **embedding** converts each word index into a **dense vector** that captures **semantic meaning**.

Example:

"outage" → [0.81, -0.23, 0.67, ...]

"emergency" → [0.85, -0.29, 0.72, ...]

📌 Similar words → similar vectors
📌 Negative words → similar direction in vector space

Example Embedding Output

Sentence →

[

[0.12, -0.45, 0.78], ← power

[0.81, -0.23, 0.67], ← outage

[0.05, 0.02, 0.01], ← in

...

]

Now the sentence is a **numerical matrix**

👉 This is what LSTM actually reads

5️⃣ Step 4: How LSTM Understands Sentiment

🔁 LSTM Reads One Word at a Time (Sequence)

LSTM processes the sequence **in order**:

power → outage → area → since → 10 → hours

At each step, LSTM:

- Remembers **important past words**
- Forgets **irrelevant words**
- Updates its internal **memory state**

Example Understanding

Word	LSTM Effect
power	neutral
outage	❗ negative signal
since	context
10 hours	❗ increases severity

📌 LSTM **learns patterns**, not rules

📌 It sees that **long outage duration → negative sentiment**

6️⃣ Step 5: Numerical Sentiment Score Generation

After processing the full sentence, LSTM outputs a **numerical score**.

Example:

Output = -0.87

Score Meaning

Score Range	Meaning
-1.0 to -0.6	Strong Negative
-0.6 to -0.2	Negative
-0.2 to 0.2	Neutral
0.2 to 0.6	Positive
0.6 to 1.0	Strong Positive

So:

-0.87 → Strong Negative

📌 This is how **sentiment becomes a number**

7️⃣ How Urgency Is Derived from Numerical Value

Sentiment alone is not enough.
So we combine:

Urgency Score =

Sentiment Strength

+ Keyword Weight

+ Context Weight

Example

Factor	Value
Sentiment score	0.87
Keyword: "outage"	+0.10

Factor	Value
Duration: "10 hours"	+0.05

Final Urgency = 0.87 + 0.10 + 0.05 = 1.02 → normalized to 1.0

📌 This is why your dataset has:

urgency_score = 0.91, 0.88, etc.

8️⃣ Why LSTM Gives the Correct Output

✅ Reason 1: Training on Thousands of Examples

LSTM sees patterns like:

- "outage + hours" → negative
- "resolved quickly" → positive
- "sparking near house" → high urgency

✅ Reason 2: Memory Mechanism

LSTM remembers **earlier important words**:

"No power since last night but now resolved"

Final sentiment → **Neutral**, not Negative

✅ Reason 3: Backpropagation

Wrong predictions → weights adjusted
Correct predictions → reinforced

Over time:
👉 Model learns **real complaint behavior**

📜 ONE-LINE VIVA / REVIEW ANSWER (MEMORIZE)

"The complaint text is converted into numerical embeddings, processed sequentially by an LSTM which captures context and severity, and finally produces a numerical sentiment score that is mapped to urgency and priority."