Deep Learning Lab 8 Report

Shusrith S

PES1UG22AM155

March 23, 2025

1 English to Spanish translation

The objective of this assignment was to perform neural machine translation from english to any language. Spanish is the chosen language for translation, because of its structural similarity to english, usage of the same script and since a large number of english words have been derived from Spanish. All these factors make it a relatively simpler language to translate to from english. This focus of this assignment is on LSTM models, which can capture some complex patterns among the data, but not enough to translate more complex languages which have different structure and syntax to input language.

2 Preparing the Data

2.1 Cleaning

From the given dataset, the first two columns containing the english and spanish texts were extracted and stored in a pandas dataframe. From there, regex was used to remove punctuation and other special characters from both the languages. For spanish, diacritic characters such as `é` were retained, since they are an important part of the language.

2.2 Tokenization

Two different types of tokenization was used for comparison purposes, namely word level tokenization and byte pair encoding. Before tokenization, in both cases, `<SOS>` and `<EOS>` tokens were added at the start and end of the sequence respectively, to indicate start-of-sequence and end-of-sequence.

2.2.1 Word level tokenization

In case of word level tokenization, all unique words were extracted from the entire corpus and based on their frequency, they were given a token value. Higher frequency words are given a lower token value and vice versa. Both languages were tokenized separately and Tensorflow's built in Tokenizer was used for this purpose.

2.2.2 Byte Pair Encoding

Byte pair encoding is a subword tokenization technique that breaks words into subwords. The algorithm starts with just 26 alphabets from the language and based on how frequently these alphabets co-occur in the corpus, the most frequent pairs are merged together to create subwords. The algorithm reaches convergence when no more such pairs can be merged together. Two forms of subword tokenization have been tried out, one where english and spanish were both tokenized separately and one where the corpus of both languages was combined and tokenized together. Separate tokenization had a vocabulary size of 14,500 for English and 27,300 for spanish. Combined tokenization had a combined vocabulary of 35,6000. Combined tokenization proved to show better results.

2.3 Padding and sequence length

2.3.1 Word level tokenization

The longest sequence in english was 47 tokens long and in spanish was 49 tokens long. A direct padding is applied to this data, where each english sequence has 47 tokens and each spanish sequence has 49 tokens. 0 is used as the padding token.

2.3.2 Byte Pair Encoding

In both cases of BPE, a token length frequency was printed and inspected, to see how many sentences have a given length. Based on visual inspection, 18 was chosen as the token length because almost 99% of input samples had less than 18 tokens. For sequences which had less than 18 tokens, `<PAD>` token was added with a token value of 0. For sequences with more than 18 tokens, chunks of 18 tokens were created and each chunk was treated as a separate input sample. For example, if a sequence had 39 tokens, three input sequences were created, two having 18 tokens each and the last one having the 3 final tokens and rest being padding.

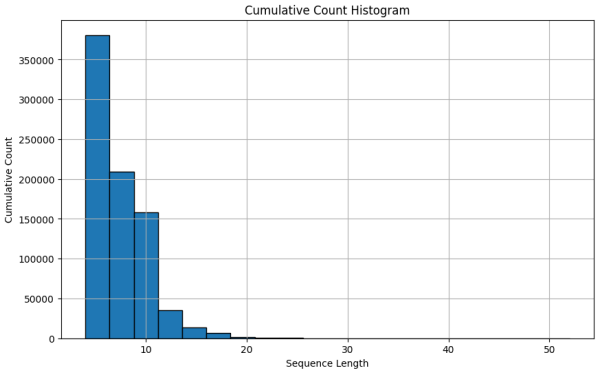

Figure 1: Distribution showing number of samples vs token length

2.4 Splitting Data

The final data had 130,000 samples. The dataset was shuffled since the smaller sequences were at the beginning and longer sequences were at the end. After shuffling, the first 100,000 tokens were used for training and the last 30,000 were used for validation and testing.

3 The Model

An LSTM-based encoder-decoder model is used for all experiments.

3.1 Without Attention

The encoder consists of an embedding layer to generate embeddings for all tokens. Since no attention mechanism is used, more memory is available, allowing for an embedding dimension of 512. The encoder is a stacked 2-layer LSTM with a dropout of 0.3 between the layers. The hidden state dimension for the encoder is set to 64.

The decoder follows the same architecture, featuring a 512-dimensional embedding layer and a 64-dimensional hidden layer with a stacked LSTM. A fully connected layer, with a size equal to the vocabulary size, is added on top of the LSTM to predict the next token in the sequence.

3.2 With Attention

When Bahdanau attention is applied, the encoder-decoder architecture remains largely the same. However, after encoding the input, the encoder output and decoder hidden state are passed through the attention module.

The Bahdanau attention module consists of three key components:

1. A layer to process the decoder's previous hidden state.
2. A layer to process the encoder's output.
3. A scoring layer to compute attention scores.

The attention scores are calculated based on the sum of the transformed decoder hidden state and encoder output. These scores determine the weighted focus on different encoder time steps.

The attention weights are obtained by applying a softmax over these scores, and a context vector is generated by multiplying the attention weights with the encoder output. The decoder then concatenates this context vector with the embedding of its current input and processes it through the LSTM.

Since additional parameters are introduced with attention, memory constraints require reducing the embedding dimension to 256, while the hidden dimension remains 64.

4 Training

In all, for the purpose of comparison, 4 different models were trained.

- No attention over byte pair tokenization
- Attention over word level tokenization
- Attention over byte pair tokenization
- Attention with bidirectional LSTM over byte pair tokenization

Batch sizes were adjusted based on GPU VRAM availability, as the number of trainable parameters varied across models. The batch sizes used were 512, 160, 512, and 256, respectively. The word-level tokenization model required a reduced batch size due to its long sequence length (47 tokens). Similarly, the bidirectional LSTM model had double the number of parameters, necessitating a smaller batch size.

A learning rate of 0.005 was used across all models, along with a learning rate scheduler that reduced the learning rate by half if the validation loss did not improve for three consecutive epochs

Cross Entropy is the loss function used. Cross entropy needs to be specifically modified to ensure that it ignores the padding token, otherwise loss will not reduce sufficiently, since majority of the sequences have a very large number of tokens. Even while measuring accuracy, since most of the tokens are pad tokens, the model will just keep predicting these pad tokens, showing a falsely high accuracy.

While calculating accuracy and loss, pad tokens need to be ignored to find out how well the model is truly learning. Even otherwise, accuracy is not a good metric due to the existence of synonyms and variablity in the structure of the sentence. It is entirely possible that the model has learnt to predict sequences purely based on context which will not match the target sequence but have the same meaning. This cannot be measure by accuracy, hence it is not a suitable metric to be used in this task.

4.1 Training Procedure

The `Se2Seq` class was implemented to integrate both the encoder and decoder. The input sequence is processed in full by the encoder, while the decoder operates **one token at a time**, starting with the `<SOS>` token. For each step, the decoder receives:

1. The previous hidden states.
2. The last predicted token (or the actual token during teacher forcing).
3. (If applicable) Attention over the encoder outputs.

The decoder then processes this information through its layers and outputs a probability distribution for the next token. The predicted token is selected using **argmax** and is used as input for the next decoding step.

4.2 Teacher Forcing

Teacher forcing was applied to all models to accelerate convergence. If the decoder were to rely entirely on its previous predictions, errors would accumulate, making it harder for the model to recover from mistakes. To mitigate this, teacher forcing was used with a probability of 0.5.

- 50% of the time, the decoder's input is its previously predicted token.
- 50% of the time, the actual target token is provided as input.

This strategy helps the model learn more effectively by maintaining a balance between self-generated predictions and ground-truth supervision.

5 Evaluation

5.1 Metrics used

For any NLP task in general and for the task of machine translation specifically, regular metrics like accuracy do not show how well the model is doing. Other metrics need to be used in order to show the actual performance of the model. The various metrics used here are :

- **BLEU (Bilingual Evaluation Understudy)**: BLEU measures the precision of n-grams in the generated text compared to a reference translation. It ranges from 0 to 100, where higher values indicate better translation quality. However, BLEU does not consider fluency or meaning preservation. A score above 30 is excellent and a score above 40 is human level translation.
- **ROUGE-1 (Recall-Oriented Understudy for Gisting Evaluation - Unigrams)**: ROUGE-1 evaluates recall based on matching unigrams (single words) between the prediction and reference. A higher score (closer to 1) indicates better word-level overlap and relevance. A rouge score above 0.4 is excellent and above 0.6 is near human level translation.
- **ROUGE-2 (Bigrams)**: ROUGE-2 extends ROUGE-1 by measuring bigram (two-word sequence) matches, providing a stronger indication of contextual coherence. Higher scores suggest better preservation of phrase-level information. A rouge score above 0.4 is excellent and above 0.6 is near human level translation.
- **TER (Translation Edit Rate)**: TER measures the number of edits (insertions, deletions, substitutions, and shifts) required to transform a model-generated translation into the reference translation. Lower TER values indicate better translations, with 0 being a perfect match. TER scores below 50 are very good and anything above 80 is not understandable.

5.2 No attention and BPE

5.2.1 Visualization

Figure 2: Loss curve for model with no attention and BPE tokenization

5.2.2 Evaluation metrics

Metric	Score
Train Loss	3.518
Val Loss	3.4703
BLEU	15.4193
ROUGE-1 (F1)	0.1958
ROUGE-2 (F1)	0.0383
TER	86.7473

Table 1: Evaluation Metrics

5.2.3 Prediction over ten random sentences

Original	Prediction	Target
i am aware of the fact	estoy estoy de	estoy informado del hecho
i hope everyone agrees	me que que de de de	confío en que toda la gente esté de acuerdo
can you lend me a dollar	puedes que un un	puedes prestarme un dólar
you may invite whoever you like	puedes que que que que	puedes invitar a quien quieras
it is just your imagination	es es un	es sólo tu imaginación
dont worry tom wont let us down	no no que tom no no a a	no se preocupe tom no nos va a decepcionar
the problem is that some of our bills havent yet been paid	el libro que es que no no no de de de de	el problema es que algunas de nuestras cuentas aún no se han pagado
we kissed in the dark	nos a la la	nos besamos en la oscuridad
you can tell tom yourself	te que que que tom	puedes decírselo a tom tú mismo
she could not cope with anxiety	ella no le que a la	ella no pudo hacer frente a la ansiedad

Table 2: 10 randomly chosen sentences from test data

The close train and val loss indicate very less overfitting. The metrics are all on the lower side, indicating that the model doesn't do well without attention. From the translated sentences too, it is visible that the model can predict the first and second token correctly, but fails to predict anything beyond that. This is mainly due to the lack of attention, where the model cannot retain contextual information beyond the initial few tokens. This model cannot be used for translation tasks.

5.3 Attention with word level tokenization

5.3.1 Visualization

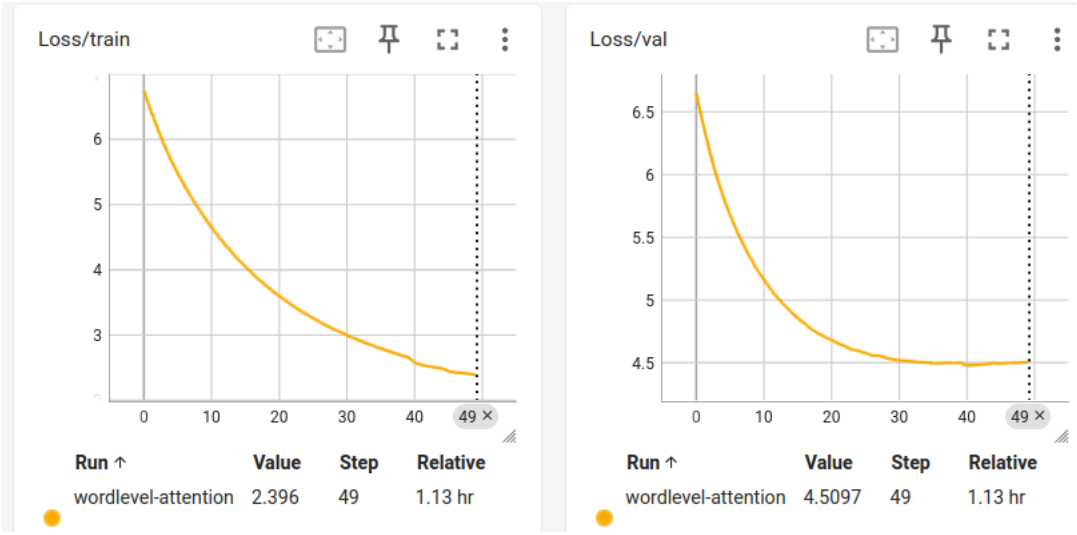

Figure 3: Loss curves for model with attention and word level tokenization

5.3.2 Evaluation metrics

Metric	Score
Train Loss	2.396
Val Loss	4.5097
BLEU	2.2642
ROUGE-1 (F1)	0.2399
ROUGE-2 (F1)	0.0299
TER	690.0192

Table 3: Evaluation Metrics

5.3.3 Prediction over ten random sentences

Original	Prediction	Target
Good to see you Tom	si me ver tom tom tom...	Qué bueno verte Tom
I thought Tom would get here ahead of us	nos que me aquí de nosotros...	Pensé que Tom llegaría aquí antes que nosotros
He asked his friends for help	la que a ayuda ayuda sus...	Él le pidió ayuda a sus amigos
Tom was a tank commander	fue un de con guerra abril...	Tom fue un comandante de tanques
She doesn't understand you	no te lo tú lo tú lo ti...	Ella no les entiende
This box is not as big as that one	caja no tan como grande mi...	Esta caja no es tan grande como esa
We should go	ir ir ir ayudar ir caza...	Deberíamos irnos
That's not my wife	que es mi esposa mi padre...	Esa no es mi esposa
Who taught you that	se que eso eso eso eso...	Quién te ha enseñado eso
Tom says he doesn't think it's possible to do that	dice que no que eso hacer...	Tom dice que no cree que sea posible hacer eso

Table 4: 10 randomly chosen sentences from test data

This model has been trained to show the importance of the tokenization technique. The word level tokenization simply tokenizes based on frequent words and uses a very large sequence of length 47. As a result, the model is glaringly overfitting, evident due to the large difference in train and val loss. The metrics are also very poor, despite the use of attention, highlighting the importance of the tokeninzation method. Despite having the same architecture as the next model, it performs very poorly.

5.4 Unidirectional LSTM with attention and BPE

5.4.1 Visualization

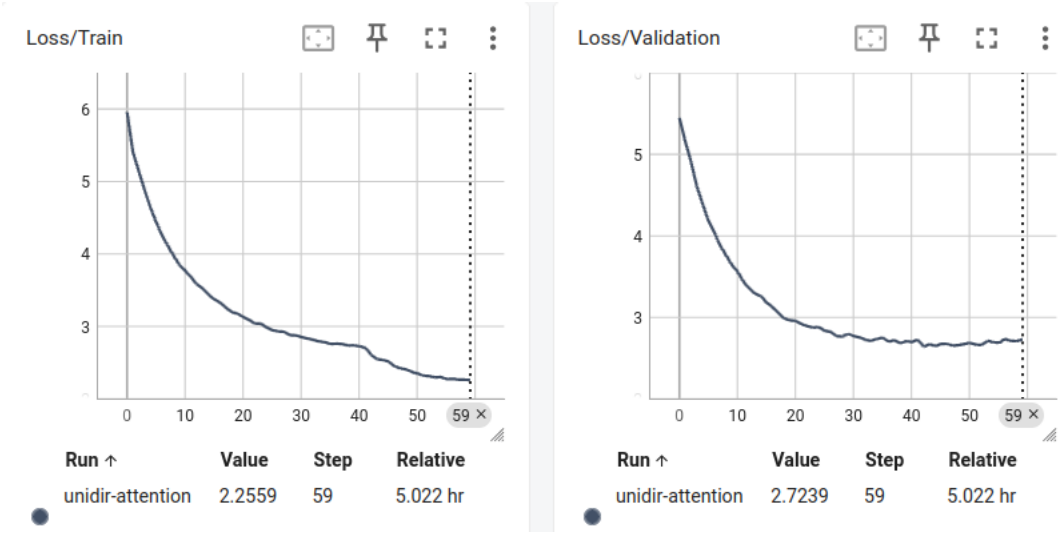

Figure 4: Loss curve for model with unidirectional LSTM, attention and BPE

5.4.2 Evaluation Metrics

Metric	Score
Train Loss	2.2259
Val Loss	2.7239
BLEU	23.856
ROUGE-1 (F1)	0.457
ROUGE-2 (F1)	0.258
TER	58.773

Table 5: Evaluation Metrics

5.4.3 Prediction over ten random sentences

Original	Prediction	Target
I should read the book	Debería leer el libro	Debería leer el libro
Please drop me off at the station	Dime por la estación por favor	Déjeme en la estación por favor
Tom doesn't remember doing what everybody says he did	Tom no se de lo lo que todo lo que	Tom no recuerda haber hecho lo que todos dicen que hizo
We don't care what he does	No te importa lo que él	No nos importa lo que él haga
Tell the truth	Dile la verdad	Decí la verdad
Tom won't be able to understand any of this	Tom no podrá entender nada de esto	Tom no va a poder entender nada de esto
I seldom go out on Monday	Normalmente voy a el lunes	Raramente salgo los lunes
I didn't want to disturb the patients	No quería escuchar a los los	No quería molestar a los pacientes
I can be your best friend or your worst enemy	Puedo ser tu mejor amigo o tu peor enemigo	Puedo ser tu mejor amigo o tu peor enemigo
Having finished it, he went to bed	Se terminado de que se a a la cama	Después de haberla terminado se fue a la cama

Table 6: 10 randomly chosen sentences from test data

This model shows a large improvement from the previous models. It has a much lower loss and shows only a very slight indication of overfitting. The metrics are much better, but there is some scope for improvement. The TER and Rouge-2 scores are not high enough for real world usage. But this model shows how effective the attention mechanism is, highlighted by the example sentences. A lot of the predicted sentences are very similar to the target sentence, and in cases where completely different words are used, when translated, they are synonyms of the actual target word. This shows that the model has actually understood the context and is able to use synonyms of words in the translation.

For example, in the last example sentence, the english sentence is `Having finished it, he went to bed` which is predicted as `Se terminado de que se a a la cama`. This sentence translates to `Now its over that he went to bed`. We can see that despite being grammatically incorrect, the predicted sentence actually preserves the meaning of the input and the attention mechanism, despite using other words, give us the same contextual information. This shows that model has learnt the data well.

5.5 Bidirectional LSTM with attention and BPE

5.5.1 Visualization

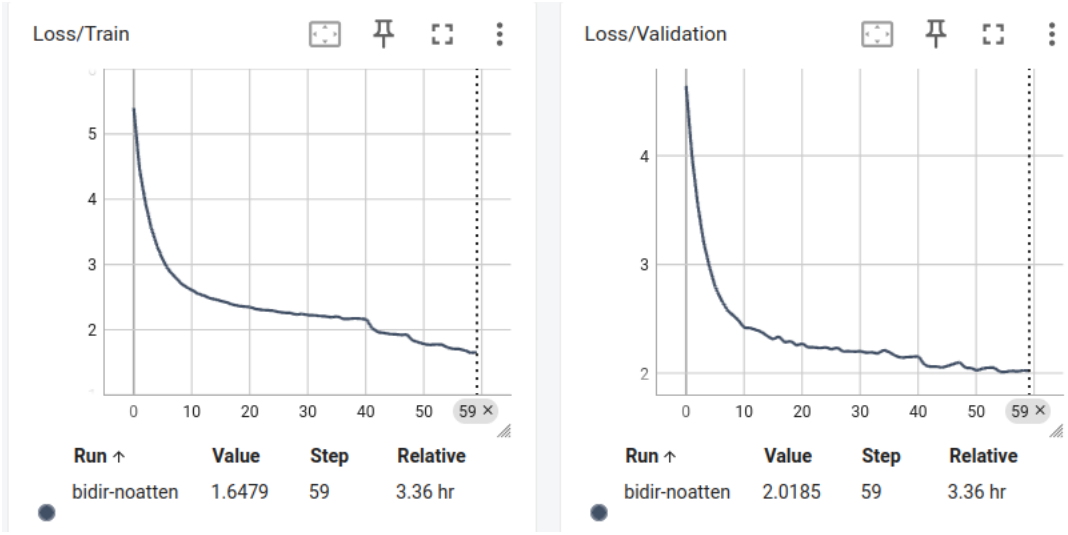

Figure 5: Loss curves for a model with bidirectional LSTM, attention and BPE

5.5.2 Evaluation Metrics

Metric	Score
Train Loss	1.6479
Val Loss	2.0185
BLEU	27.68
ROUGE-1 F1	0.540
ROUGE-2 F1	0.371
TER	49.03

Table 7: Translation Evaluation Metrics

5.5.3 Prediction over ten random sentences

Original	Prediction	Target
So Tom what can I do for you today	Así que Tom qué puedo hacer por hoy	Así que Tom qué puedo hacer hoy por ti
He has a camera	Él tiene una cámara	Él tiene una cámara
Everybody admired his courage	Todos los mundo su su valor	Todo el mundo admiraba su coraje
I haven't found my keys yet	Todavía no he encontrado mis llaves	Todavía no he encontrado mis llaves
Where exactly is Tom	Dónde está exactamente Tom	Dónde está exactamente Tom
What are you going to see	Qué vas a ver	Qué vais a ver
I don't know if you'll be here when I return	No sé si estarás aquí cuando vuelva	No sé si estarás aquí cuando vuelva
I'm going to tell you everything	Te voy a decir todo	Te lo voy a contar todo
Do you snore	Tú	Tú roncas
I love kids	Me encantan los niños	Me encantan los niños

Table 8: 10 randomly chosen sentences from test data

This model shows the best performance, showing a very low level of overfitting and very good evaluation metrics. The translated sentences show that its predicting near perfect sentences, not even using synonyms this time and retaining the correct grammatical syntax. Adding the bidirection to the LSTM enables the model to extract more context from the input, hence it leads to much improved performance.

6 Visualizing attention weights

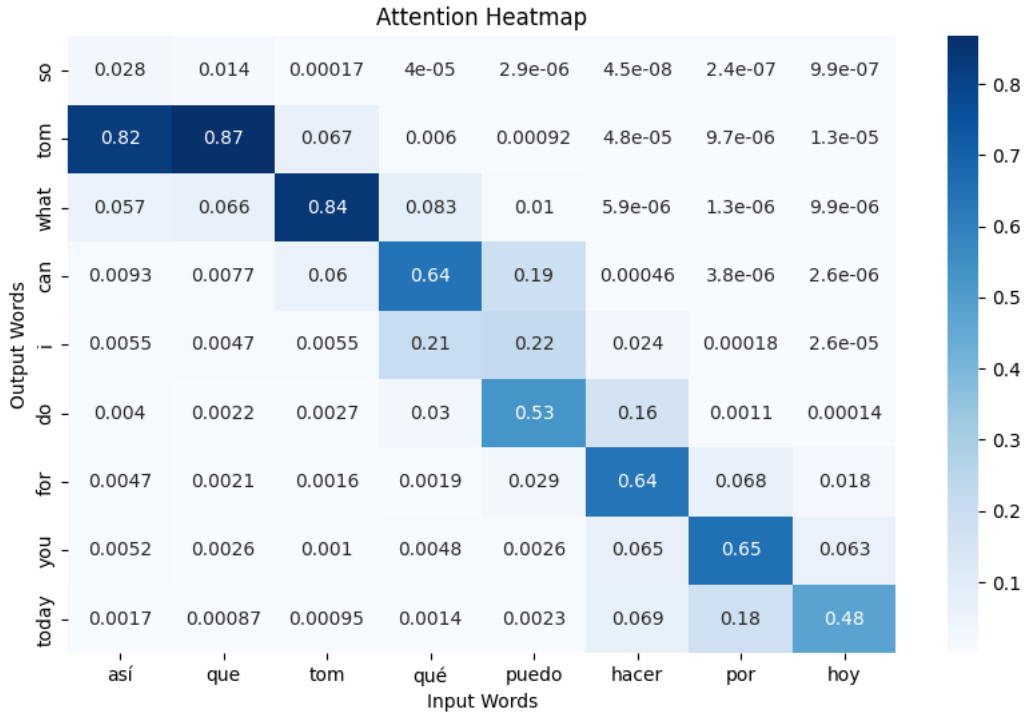

Figure 6: Visualization of attention for a test sentence

Input sentence : `So Tom, what can I do for you today`

Predicted sentence : `Ası que Tom que puedo hacer por hoy`

Inspecting the attention weights, we can see that

- **"Tom" (output) has a strong focus (0.98) on "tom" (input)** → This means the model correctly aligns "Tom" in English with "tom" in Spanish.
- **"What" (output) is strongly linked to "qué" (input) with weights 0.82 and 0.87** → This is correct since "qué" means "what."
- **"Can" (output) has high attention (0.84) to "puedo" (input)** → This is also correct, as "puedo" means "can" in Spanish.
- **"Do" (output) has a strong link to "hacer" (0.64)** → This makes sense since "hacer" means "to do/make" in Spanish.
- **"For" (output) is linked to "por" (0.53)** → This is reasonable because "por" can mean "for" in some contexts.
- **"Today" (output) aligns with "hoy" (0.65)** → This is correct, as "hoy" means "today."

Overall, the attention alignment shows that the model captures key word correspondences well, particularly for high-confidence words like *Tom*, *qué*, and *puedo*. However, some minor misalignments, such as *for* and *you*, suggest areas for improvement. Further fine-tuning or enhanced attention mechanisms could help refine these mappings for better translation quality, but overall, the model is doing reasonably well.