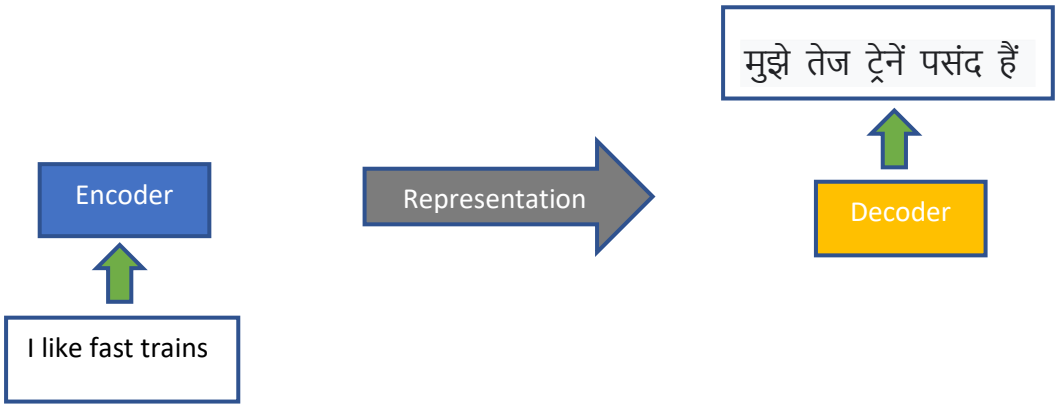
Self Attention… The way I Understand

Part 1

Transformers … State-of-the-art deep learning architecture for NLP

Transformer is replacing the RNN and LSTM in the world of Natural Language Processing. Many of the modern NLP models such as GPT, T5 and BERT are based on the transformer architecture. The traditional RNN/LSTM based approach fail to learn the long term dependencies among words in a text and this prevented achieving good NLP models beyond a point. With the oncoming of the transformer, the field of NLP witnessed an breakthrough inform of new architectures such as GPT-3, BERT and others.

Transformer consists of an encoder-decoder blocks. The input sentence is fed to the encoder. The encoder transforms the sentence into a representation. The internal representation of the words in the input sentence is like a hidden state which is passed on to the decoder bloc. The decoder decodes the representation to generate the output. Suppose we need to build transformer based model for translation from English to Hindi. The block level architecture for this model would look like –



To get a conceptual understanding of how the Transformer models translate, we need to understand the concept of Self Attention. To understand the concept of Self Attention, it is paramount that we understand some important aspects of a language.

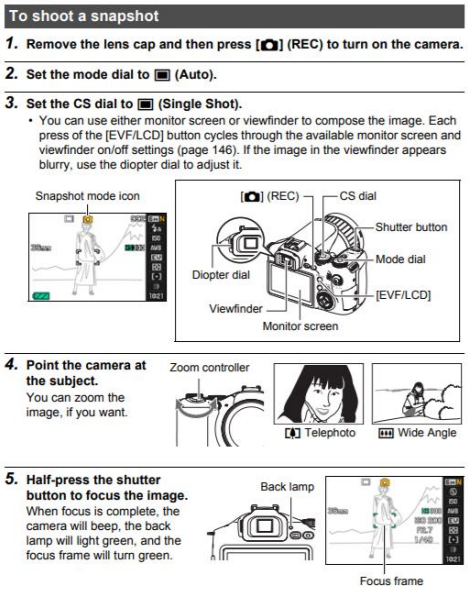
Language As a Tool for Information Transfer

Language is a tool for communication / exchange of information. It consists of tokens and rules of using those tokens (grammar of the language). Using the tokens in the right manner (as defined by the grammar) we communicate and exchange information as I am attempting to do now. The string of tokens that you have read till now and the grammatical rules I followed, hopefully would have passed on to you what I wanted to communicate till this point. So, what has happened based on the lines you have read till here is passing of an idea from my mind to yours.

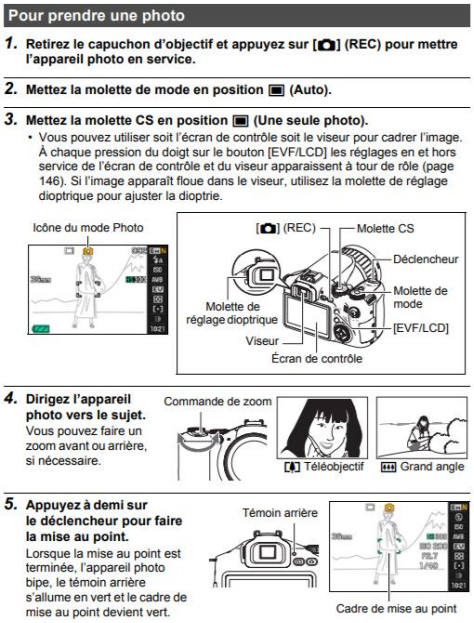
The same information could have been passed on to you in some other language such as Hindi, Marathi and Tamil etc. Assuming you know the language, you will get the same information as what

you got when I used the English language. Thus, information is independent of the language used to transfer it. When we purchase a digital device such as a camera, we get a manual which has instructions about the device in various languages. The manual has a list of topics covering various aspects of how to handle and use the device. For example, the following two images are of pages from an instruction manual to take snapshot from a CASIO Ex-F1.

Ref. ://support.casio.com/fr/manual/manualfile.php?cid=001004010



English



French

Whether the instructions are read in English or in French or any other language, the result will be the same. We learn how to shoot a snapshot using this camera. The authors of these manuals have communicated same information using words and grammar of respective language to transfer the information to the reader. **That means the information exchanged between the author and reader is independent of the language used.**

The exchange of information through the user manual helps the reader gets to know how to handle and use the various aspects of the camera. She moves from a state where she had no idea on how to operate the camera to a state where she can now take snapshots. She has gained some knowledge which helped get clarity on how to use the camera.

Quantifying Information

The **degree of clarity that results from exchange of information can be quantified using formulas such as Gini and Entropy. This is done by Decision Tree algorithm** where the information gained manifests as the branches connecting the parent to child nodes and the branching function with thresholds. **In DNN based techniques, the same quantification is done in other ways such as using the Sigmoid and Tanh function in the "Input Gate" of LSTM**. Though, unlike Decision Tree, DNN based models are black box models. (pl. refer: https://www.linkedin.com/posts/mukeshrao1_rnn-

intuitive-way-activity-6833251434050539520-GSQR). The information quantification methods associate the degree of clarity with a quantity. They are not concerned with what exactly the information is. Using any quantification method one can compare two or more sentences to decide which one gives more clarity. For example, “Raj had little sugar”. This statement conveys some information about Raj but is ambiguous. Does it mean he had mild blood sugar problem, or does it mean he ate a little sugar? Instead, if the sentence is “Raj had mild diabetes”, we immediately get clear information about Raj. The second sentence gives more clarity and hence conveys a higher quantum of information than the first one. It is easy for us to associate a quantum with the given sentences to qualify whether it is meaningful or not. How do we train a computer system to evaluate the sentences? How do we train them to decide whether the combination of words is meaningful or ambiguous? This is where a corpus plays an important role.

Corpus for Language Modelling

Corpus is a massive collection of written text or speech that represents the way a language is used in communicating information. It contains instances of correct use of a language in a particular domain (operating digital cameras) consisting of various topics relevant to that domain (“Taking a snapshot”, “Cleaning the lens”, “Charging the camera” etc. There are plenty of open source corpora in multiple languages available on the net. (Ref: https://en.wikipedia.org/wiki/List_of_text_corpora).

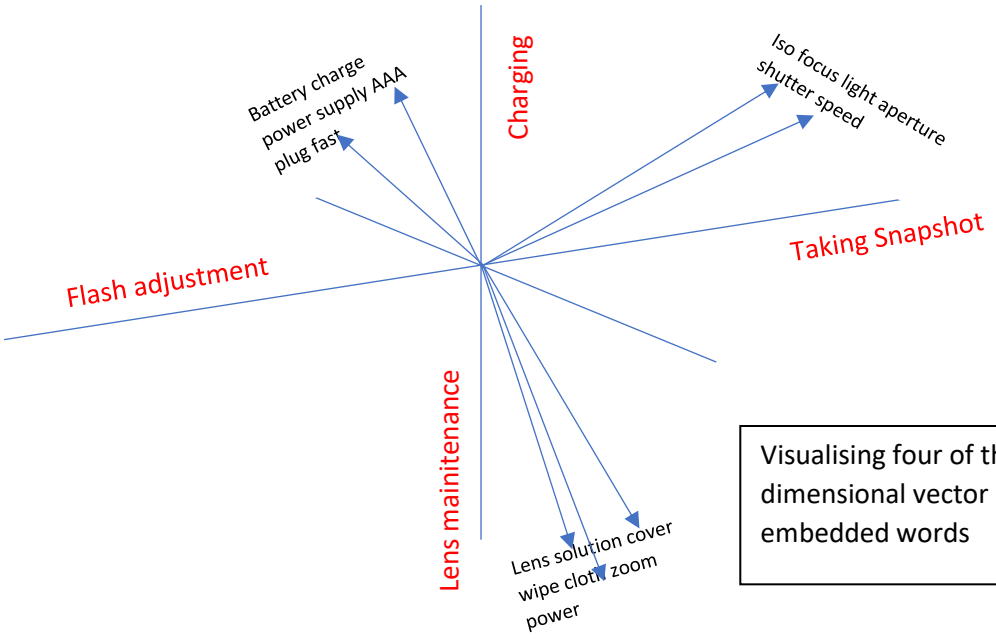
Imagine you have a huge collection of the manuals explaining how the CASIO digital cameras operate. The instructions are available in multiple languages. This collection can be considered as a language wise corpus that explains how to use the CASIO digital cameras.

Across all the manuals on operating the CASIO digital camera, many of the topics will be the same. In each topic combination of words used will be such that they convey information with maximum clarity. For example, under “How to take a snapshot”, the words could be about lighting conditions, selecting ISO etc. We may not find reference to words like “battery” or the “camera bag” in this topic. The probability of finding these words in the context of “Taking a snapshot” is relatively low because they have no direct purpose in that topic and would hamper clarity if used. For example, “To take a clear snapshot the camera bag should be closed”. Does this make sense? The sentence is grammatically correct but leaves one scratching the head. It creates ambiguity not clarity and hence should get a low information quantum value. Corpuses are supposed to represent correct use of combination of words that deliver maximum clarity.

In this example of corpus of digital camera operation, we know the likely topics and likely use of words in each topic. However, when dealing with large corpuses we usually do not have information on the number of topics hidden in the corpus. We must explore and extract the likely topics.

Topics Identification in a Corpus

Using topic identification tools such as LDA (Latent Dirichlet Allocation) we can get an estimated number of logical topics in the corpus. The number of discovered topics would be like the actual number of topics in the corpus with similar meaning as the actual topics in the manual. These topics can be considered as dimensions of a vector space in which we would like to embed the words using techniques such as Word2vec or Glove.



Visualising four of the d dimensional vector space with embedded words

Embedding words the magic behind NLP

The embedded word vectors will form natural clusters in that d dimensional vector space (d = number of topics). The word clusters may not be well formed but their location in the vector space will reflects their importance in the context of the topics. This is where the magic happens. Magic that makes language modelling possible.

The embedded word vectors stored elementwise in an array (call it X) contain all the information required to create a language model. Information such as which words correlate with which other words and how much. Quantum of information carried in the context of the various dimensions. For example, let one of the sentences in an imaginary corpus be "How are you". Assume the words in this corpus are embedded in three dimensions (d = 3). The value of d could be any number irrespective of number of words.

Word Embedding X

	d1	d2	d3
how	12	7	2
are	10	9	5
you	17	10	3

This matrix is rich in information about the words, their correlations, and the quantum of information they carry in the context of the dimensions d1, d2 and d3.

Techniques such as Simple RNN and LSTM attempt to extract this information using the concept of Hidden State vector, Input Triggers and Cell State vectors. Self-Attention method extracts this information in a different way.

Self Attention building blocks – Query, Key and Values

When we form sentences using the words and corresponding grammar, one can easily see that the words have immediate sequential dependency and dependencies across the sentence. For example, in the sentence "Self-Attention topic is hot as it is almost magical in results.", what does the word "it" refers to? It is linked to the word "Self-Attention". Thus, the word "it" depends on "Self Attention" to be meaningful. Understanding this dependency between far separated words is one of the keys to effective language modelling. Any two or more words may be related to one another at any time in each sentence.

Understanding how important a word is from its own frame of reference in the context of other words in the sentence, is called "**Self Attention**". It is a method of evaluating the degree of dependency of each word on other words in a sentence. At a very abstract level, given a word (lets call it **Query**) find other words in the same sentence to which the Query word is related. Let us call the list of words returned "**Key words**". For example, from the d dimensional vector space in figure1, if the query word is "focus", the key words returned could be "shutter", "speed", "aperture" etc.

The **Query and Key words together** in the context of the topics (dimensions of the embedded space) convey some quantum of information. Let us call this quantum "**Value**". From the feature space it is clear these words will be more meaningful in the topic "taking a snapshot" compared to other topics.

Is this not like firing a database query on a table and getting the results? The word embedding array that we call "X" is like a table that holds the tree vital information which we must extract firing a query. We can represent the relation between X, Q, K and V as follows:

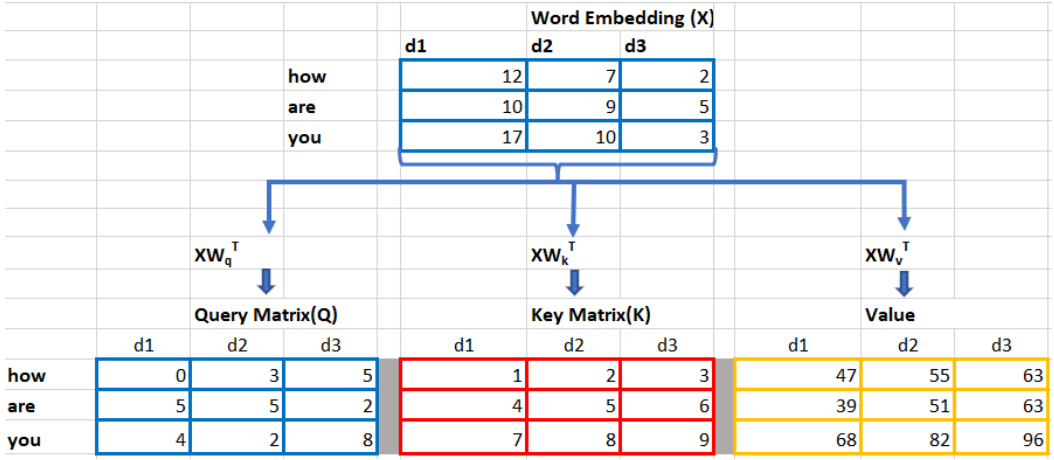
$$Q, K, V = f(X) \rightarrow 1$$

Using some function f on X, we should be able to separate out the K and V for a given Q. The function **"f" can be a Deep Neural Network (DNN) model**. DNN is known as a universal function approximator. It can represent any function as approximately as desired. However, unlike RNN / LSTM which are also neural network based algorithms and iterative (Ref: https://www.linkedin.com/posts/mukeshrao1_rnn-intuitive-way-activity-6833251434050539520-GSQR)., Self Attention mechanism does not employ iterations. Equation 1 indicates that given X we need to find a function that will result in three mathematical objects (matrices) representing Query, Key and Values quantitatively.

Note: this is not matrix decomposition of X that is done for example in Latent Dirichlet Allocation. You will not get back X by multiplying Q*K*V that is because these matrices are derived from neural network through application of corresponding weight matrices. The mechanism takes as input the entire sentence / document to process all the words in parallel without the need for any iterations over time steps.

Self Attention Mechanism

1. `**Derive the Q, K and V matrices from the X matrix. This is done by multiplying the X matrix with W_q, W_k, V_k weight matrices respectively**. These weights are learnt through back propagation during the training stage



	Word Embedding (X)		
	d1	d2	d3
how	12	7	2
are	10	9	5
you	17	10	3

XW_q^T → Query Matrix(Q)

	d1	d2	d3
how	0	3	5
are	5	5	2
you	4	2	8

XW_k^T → Key Matrix(K)

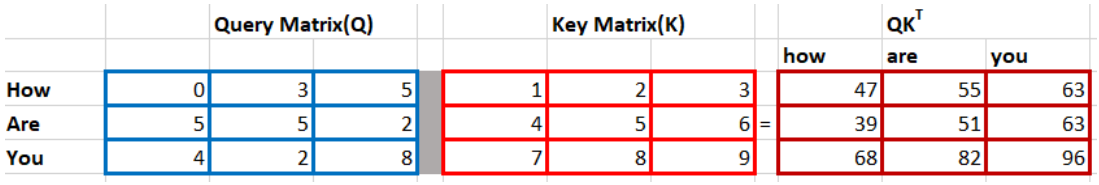
	d1	d2	d3
how	1	2	3
are	4	5	6
you	7	8	9

XW_v^T → Value

	d1	d2	d3
how	47	55	63
are	39	51	63
you	68	82	96

Note: All the numbers in all the matrices are only for example.

2. Given that the Q and K matrices are representation of the embedded words in the role of Query and Keys, each word will take on the role of a Query word and be part of Key words for other words. **Since words are embedded in d=3 dimensions, their avatar as Query and Key will also be in d=3 dimensions**
3. To extract the degree of inter word dependency in a sentence, get the dot product QK^T. The dot product returns a square matrix of shape dXd. The words vectors in proximity in the vector space will have higher dot product reflecting higher similarity
4. The elements values in the resulting QK^T matrix reflect the degree of inter word dependency in the given sentence



	Query Matrix(Q)				Key Matrix(K)				QK^T		
									how	are	you
How	0	3	5		1	2	3		47	55	63
Are	5	5	2		4	5	6	=	39	51	63
You	4	2	8		7	8	9		68	82	96

Note: QK^T dimensions will be number of words X number of words for a sentence

5. Through the dot product we get the Q and K interdependencies. Next, we need to extract quantum of information conveyed by the word combination.
6. For a technical reason the elements of dot product are divided by square root of the dimensions of the embedding space. This, I believe helps in stabilizing the gradients i.e. prevent the gradients of the error function w.r.t the weights of W, K, Q from fluctuating wildly. This is like scaling data in image processing (divide every pixel by 255) which is done for the same reason. Since d = 3 in this example

	QK^T/sqrt(3)		
	how	are	you
how	27.13546	31.75426	36.37307
are	22.51666	29.44486	36.37307
you	39.25982	47.34272	55.42563

7. The values in the QK^T/sqrt(3) are raw numbers whose magnitude indicate degree of influence one word has on the other in the sentence. To make it more useful, we convert the raw numbers into probability values using SoftMax function. The resultant matrix is called the **Score Matrix**

	Softmax(QK^T/sqrt(3))			
	how	are	you	
how	9.63506E-05	0.009767	0.990136	1
are	9.58989E-07	0.000979	0.99902	1
you	9.53114E-08	0.000309	0.999691	1

8. The element values indicate the word “how” is more strongly related to word “you” as is evident in first row values. In the second row we notice that the word “are” is also strongly related to “you”. The third row comparing “you” with “you” and obviously has the highest degree of dependency. (**Note**: the element values are only samples taken out of thin air to explain the concept)

9. Next step is to compute the **Attention Matrix**. This matrix reflects how much information a combination of words convey given the topic context/ dimensions. For this, the **Score Matrix** is multiplied with the **Value Matrix**

	Softmax(QK^T/sqrt(3))			
	how	are	you	
how	9.63506E-05	0.009767	0.990136	1
are	9.58989E-07	0.000979	0.99902	1
you	9.53114E-08	0.000309	0.999691	1

	Value		
	d1	d2	d3
how	47	55	63
are	39	51	63
you	68	82	96

10. The value matrix shows the quantum of information conveyed by a word in the context of each dimension. For example, if the topic is “How to take a snapshot”, the word “focus” is likely to be convey more information than a word like “warranty”.

11. The degree of interaction between two words such as word “how” and itself ((9.63506E-05) given in the “Score” matrix, will be multiplied with information conveyed by the word “how”

(47, 55, 63) in the context of the respective dimensions (d1, d2, d3) in "Value" matrix. The Self Attention for the word "how" will be calculated as follows–

	Softmax(QK^T/sqrt(3))			
	how	are	you	
how	9.63506E-05	0.009767	0.990136	1
are	9.58989E-07	0.000979	0.99902	1
you	9.53114E-08	0.000309	0.999691	1

	Value		
	d1	d2	d3
how	47	55	63
are	39	51	63
you	68	82	96

a. Quantum of information contributed by the Query word "how" in combination with key word "how" = 9.63506E-05 * (47,55,63)
b. Quantum of information contributed by the Query word "how" in combination with key word "are" = 0.009767 * (39, 51, 63)
c. Quantum of information contributed by the query word "how" in combination with key word "you" = 0.990136 * (68, 82, 96)
d. Self Attention for word "how" = a + b + c

12. Similarly, the Self Attention for the query word "are" is calculated as follows (shift the small ovals one row down in the score matrix)
    a. Quantum of information contributed by the Query word "are" in combination with key word "how" = 9.58989E-07 * (47,55,63)
    b. Quantum of information contributed by the Query word "are" in combination with key word "are" = 0.000979 * (39, 51, 63)
    c. Quantum of information contributed by the query word "are" in combination with key word "you" = 0.9902 * (68, 82, 96)
    d. Self Attention for word "how" = a + b + c

13. Repeat the steps to find the Self Attention of the word "you" (shift the small oval rings to the last row in Score matrix and repeat steps a to c.

	Self Attention			
	how	are	you	Z(Self Attention)
how	0.015897849	1.494398	243.5735	245.0838
are	0.000158233	0.149756	152.8501	153
you	1.57264E-05	0.047228	245.924	245.9713

Some Key Points:

I. Self Attention calculation is **NOT a dot product** between the Score and Value matrices.
II. Should not the Score matrix have value of 1 in the diagonal and the upper half (above diagonal) values be same as lower half (below the diagonal)? Should not the score for a combination of words for example ("how", "are") be the same as the score for ("are", "how")? Answer to both the questions is "Not necessary".
    a. Suppose we have a statement "Its good not bad" and "its bad not good". The words are same but the two statements have opposite polarity giving different information. Combination ("not", "bad") gives a different quantum of information than the combination ("bad", "not") which anyway seems incomplete.
    b. Diagonals represent a combination of word with itself. Such combination may or may not be valid. For example, when the word "minute" is used to represent quantity, may not make any sense to score ("minute"," minute"). The score may be close to zero as such pairing makes no sense. On the other hand, if the word "minute" refers to time then scoring a pair ("minute", "minute") makes sense as the sentence may be "minute by minute".

Attention Matrix Purpose

The final output of a Self Attention process is a "Attention Matrix". This matrix helps a trained model understand which word depends upon which other word/s in the sentence to be informative and the quantum of information conveyed. For example, in following two sentences -

I. "I did not like the movie we saw at the theatre because its storyline was not good", the word "it" points to the word "movie" not "theatre".
II. "I could not enjoy the movie we saw at theatre because it lacked proper acoustics". In this sentence the word "it" refers to the word "theatre" not the word "movie".

A trained model will have the three weight matrices Wq, Wk and Wv filled with weights learnt during the training stage. Using these weight matrices on the input sentence of embedded words X, the Q, K and V matrices are formed. The "Self Attention" matrix Z is generated (as discussed in bullets 9 to 13 above). The "Self Attention" matrix represents the encoded form of the input sentence (like the hidden states in RNN and LSTM are formed from input embedded words). **That is the purpose of Self Attention process.**

Unlike in RNN/LSTM algorithms, here the entire sentence is fed to the algorithm in one shot. **There is no recurrence in the training stage**. Feeding the words of a sentence in parallel helps reduce the training time and helps learn the far separated word interdependencies much more effectively (thanks to the self attention concept) than the sequential models can.

The Self Attention form of the input is shared with the decoder for further processing to achieve the final objective which could be translating into another language, sentiment analysis or text generation etc.

Positional Encoding

Unlike the RNN/LSTM, Self Attention mechanism does not have any iterations in the training stage. The entire input sentence is fed into the Encoder for processing. Which means, the sequential arrangement of the words in a sentence is ignored. This sequential relation is captured by RNN/LSTM but not by Self Attention mechanism we discussed. But we know the importance of the sequential relation among words. To address this problem, the original authors of the Transformer architecture (Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin), introduced a concept call positional encoding. I will discuss this in part 2 of this post.

References –

1. "Attention is all you need" - https://arxiv.org/abs/1706.03762
2. "Getting Started with Google BERT" – Sudharsan Ravichandran / Packt
3. "Python Machine Learning by Example" – Yuxi (Hayden) Liu / Packt