**Topic Modelling**

Topic modelling refers to the task of identifying topics that best describes a set of documents. These topics are not predefined but are extracted from the set of documents hence are "latent". It is an unsupervised method, it doesn’t require a predefined list of tags or training data that’s been previously classified by humans.

**Need for Topic Modelling**

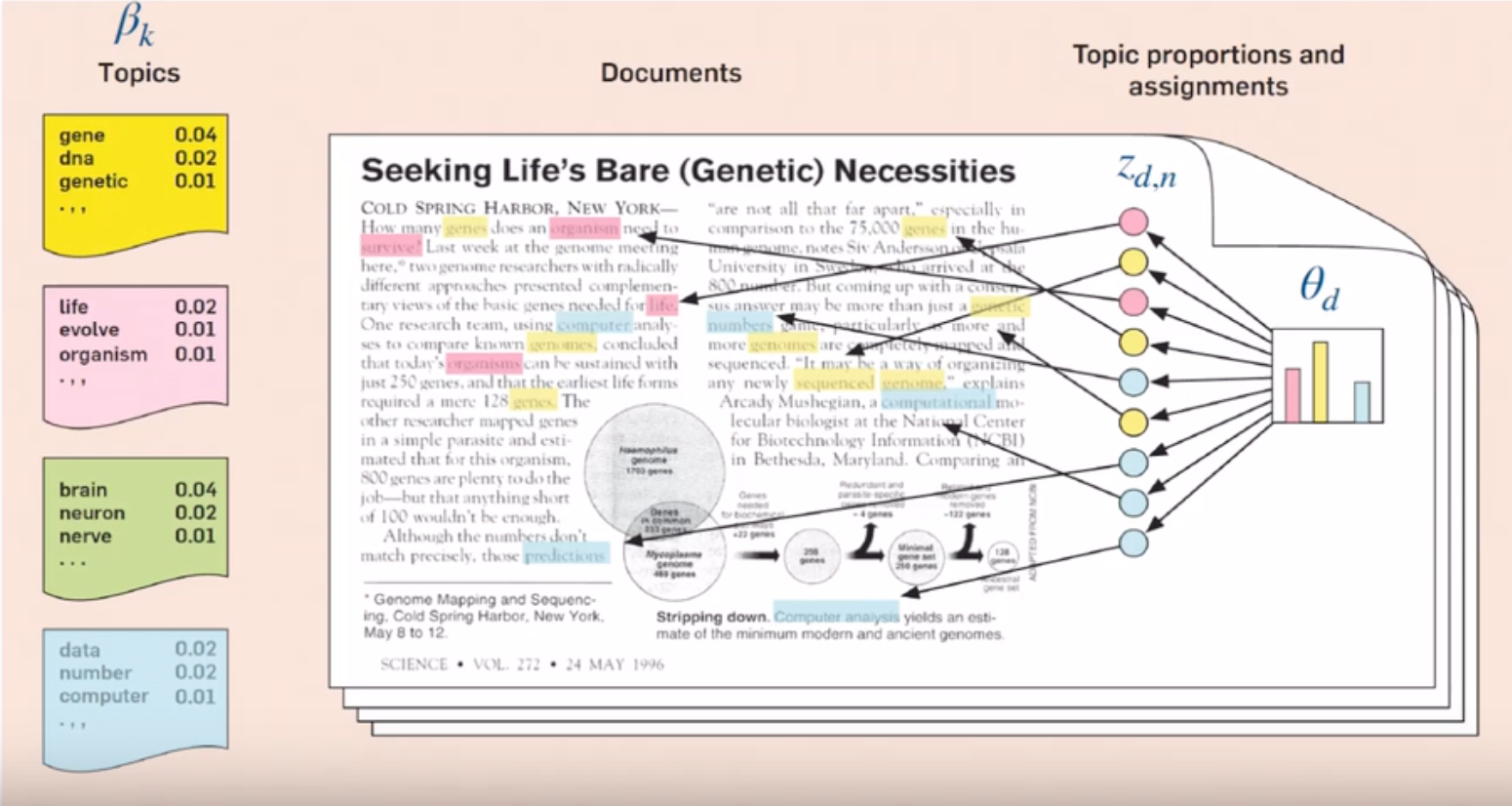
Here are few uses of topic modelling:

1. Analyzing comments/reviews online about a product or service and find out the topics people are discussing about the product. For example: Reviews about phone might have topics about display, sound, memory, processing.
2. Recommending products. Suppose if a customer likes a certain book then recommending other books with similar topics.
3. Categorizing news based on context.

etc.

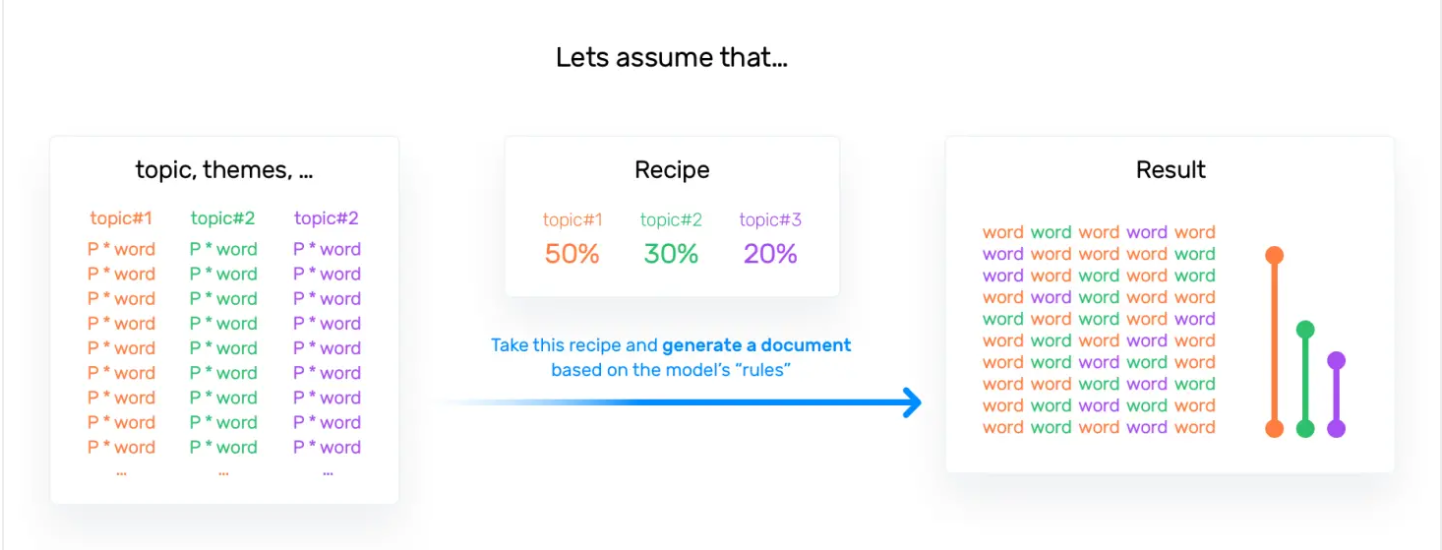
**Latent Dirichlet Allocation (LDA)**

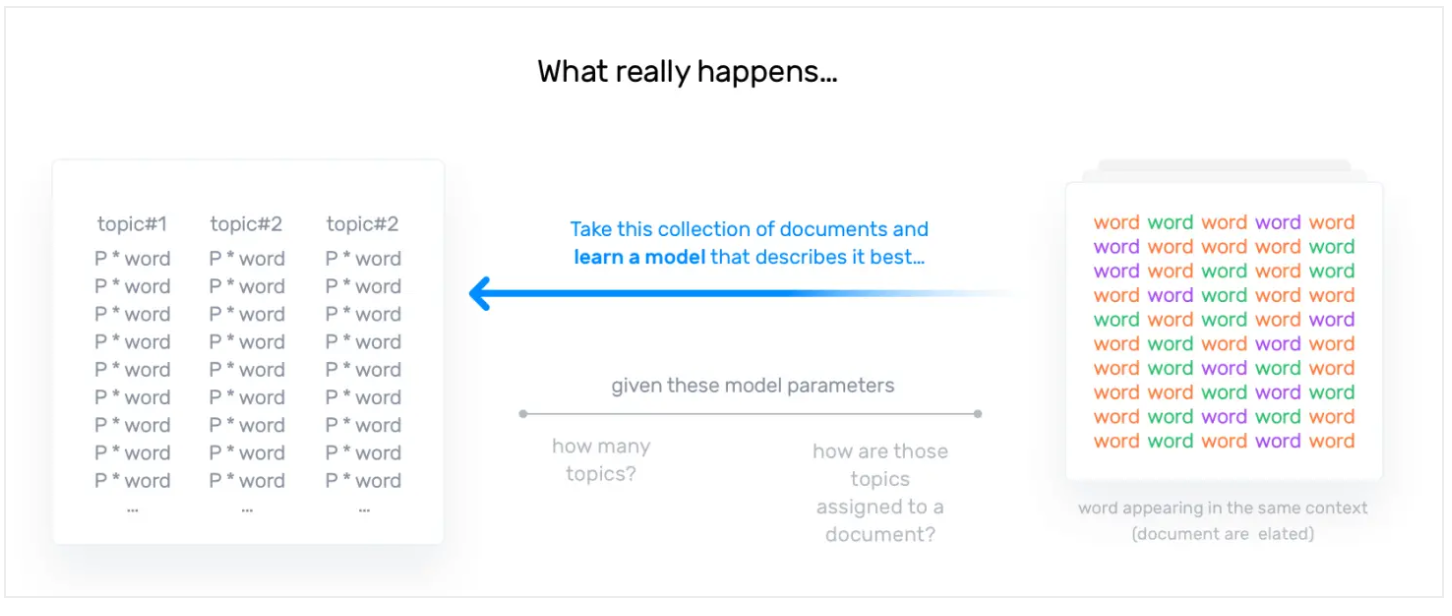
LDA imagines a fixed set of topics. Each topic represents a set of words. And the goal of LDA is to map all the documents to the topics in a way, such that the words in each document are mostly captured by those imaginary topics.



The basic idea is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words.

*“Each document can be described by a distribution of topics and each topic can be described by a distribution of words”*

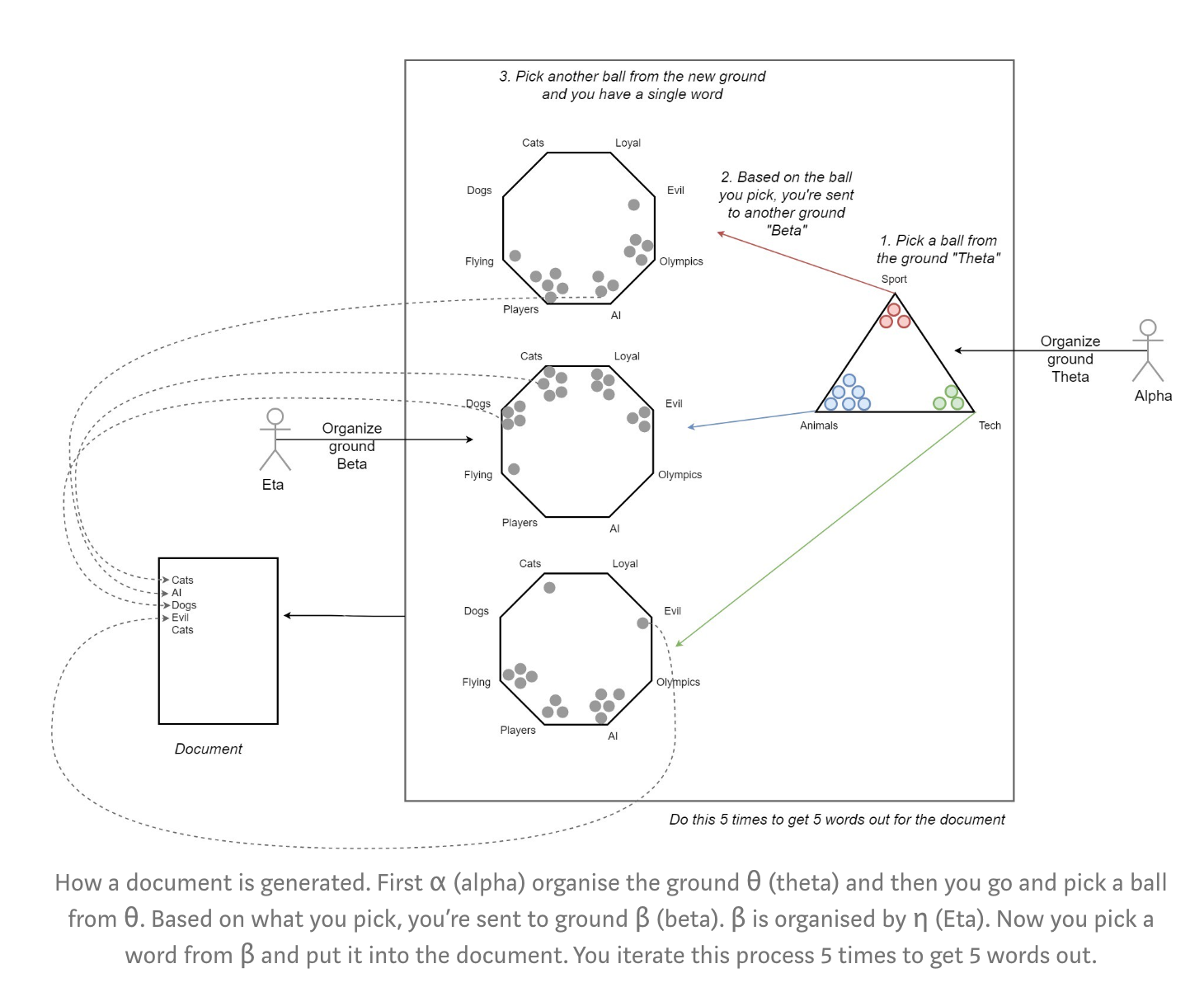




If we are given a topic distribution for a document and word distribution per topic, generating document becomes simpler. But in case of LDA we are given with document and we need to figure out topic and word distribution.

Let’s see an example:

Assume θ is topic distribution per document (**w**) and β is the word distribution per topic, now we pick a topic from θ and based on that topic pick words from β distribution and make our document. But how these θ and β are managed? These are managed by their organizer called α and η respectively. These α and η are called priors and come from “*Dirichlet distribution*”.



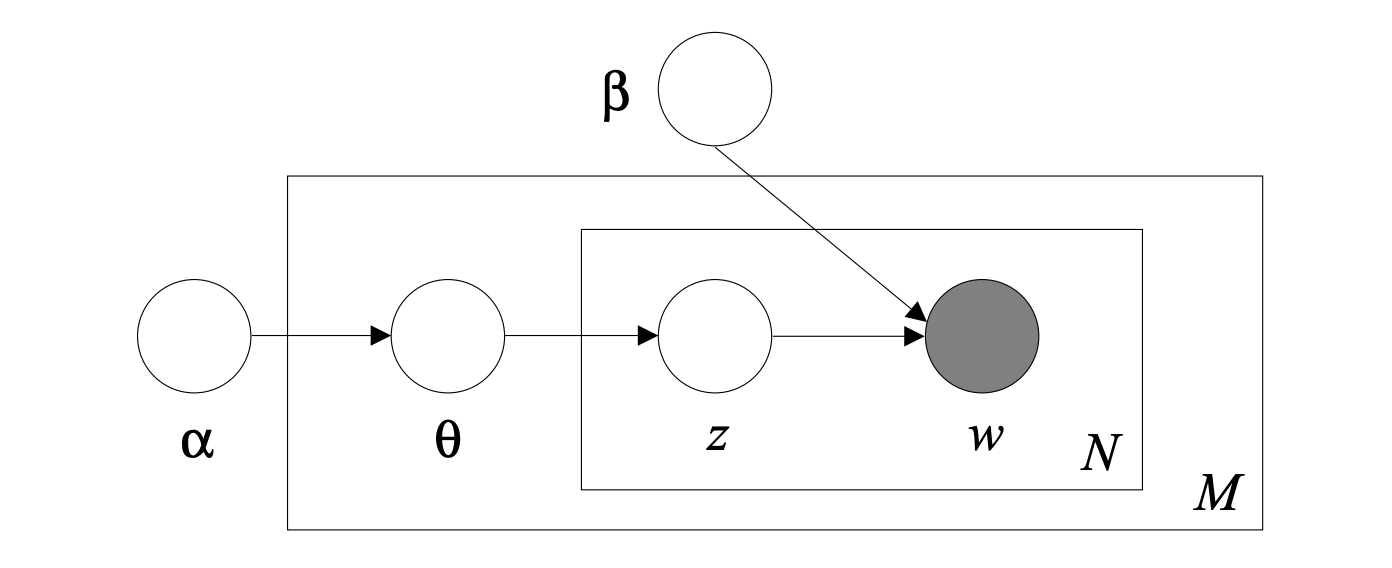
The above figure shows an already learnt LDA system, when all the latent parameters are learnt. But to arrive at this stage we need to first learn these latent parameters.

Now some mathematics and more notations:

**LDA representation**

*Definitions and notations*

* k — Number of topics a document belongs to (a fixed number)
* V — Size of the vocabulary
* M — Number of documents
* N — Number of words in each document
* *w* — A word in a document. This is represented as a one hot encoded vector of size *V* (i.e.*V* — vocabulary size)
* ***w***(bold *w*): represents a document (i.e. vector of “*w*”s) of *N* words
* *D —* Corpus, a collection of *M* document
* z — A topic from a set of *k* topic. A topic is a distribution word. For example, it might be, *Animal = (0.3 Cats, 0.4 Dogs, 0 AI, 0.2 Loyal, 0.1 Evil)*



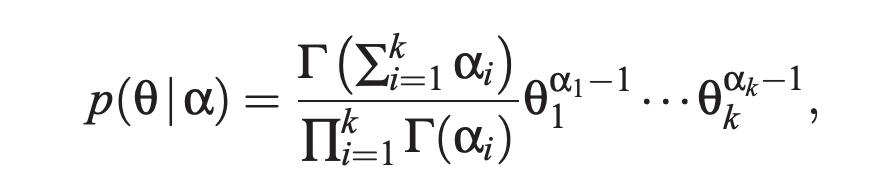
*Graphical model representation of LDA*

There are three levels in LDA representation:

**Level 1:**

The parameters α and β are corpus level parameters, assumed to be sampled once in the process of generating a corpus.

α has a topic distribution for each document (θ ground for each document). α is a vector of dim *k.*



αi > 0 and Γ(x) is gamma distribution. θi ≥ 0 and . This is Dirichlet Process.

β generates word distribution for each topic and have matrix shape (*k x V)*

*“Second, the word probabilities are parameterized by a k ×V matrix β where*

*βi, j = p(wj = 1|zi = 1), which for now we treat as a fixed quantity that is to be estimated”*

Total prior parameters: k + kV *(independent of training corpus)*

**Level 2:**

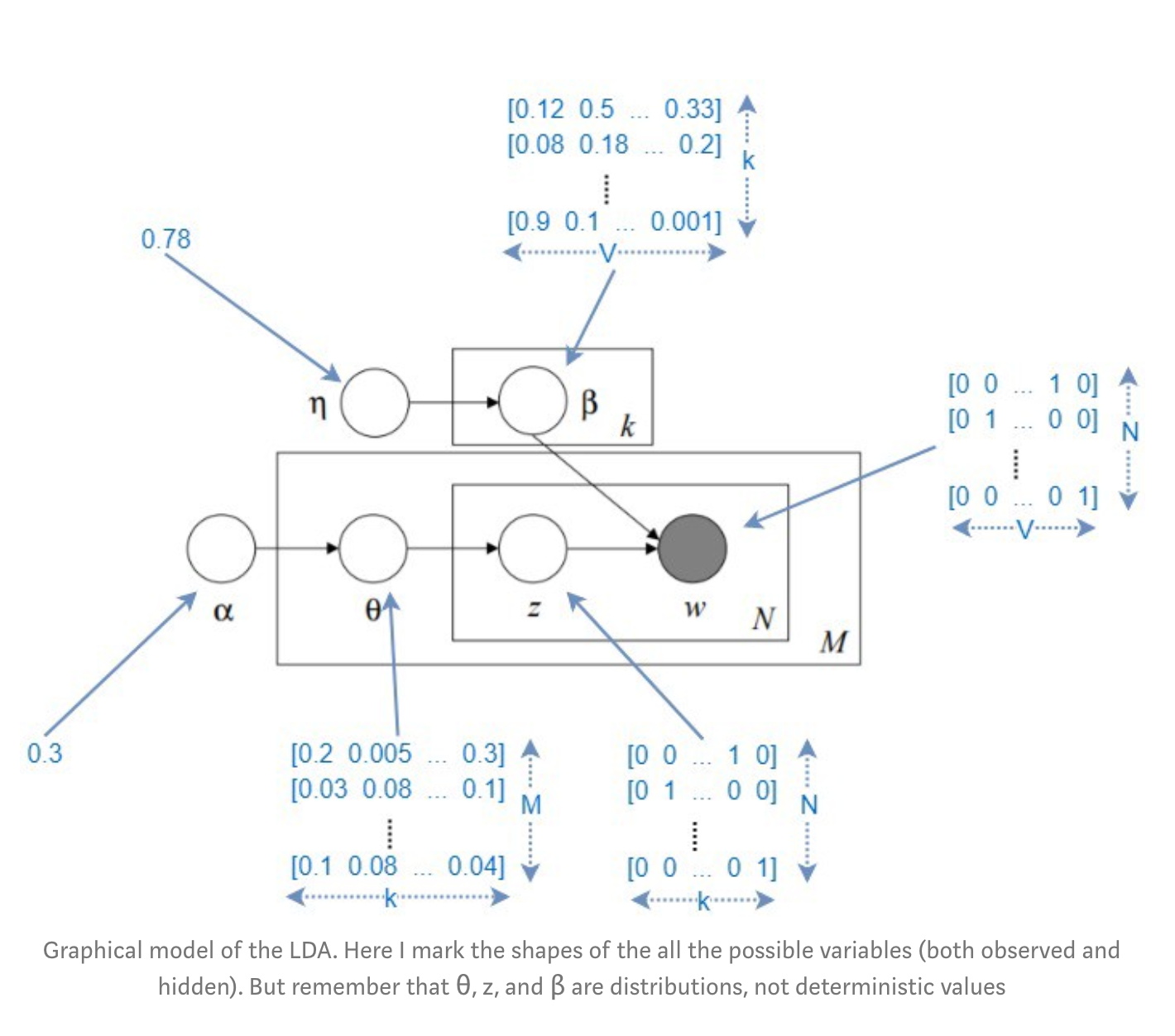
The variables θd are document-level variables, sampled once per document. For each document, a vector of shape *k*. Matrix shape: (*M x k)* where θ(i,j) represents the probability of the ith document to containing the jth topic

**Level 3:**

Finally, the variables zdn and wdn are word-level variables and are sampled once for each word in each document.

wdn : 1 to *N*

zdn : matrix shape (*N x k)* derived from θd



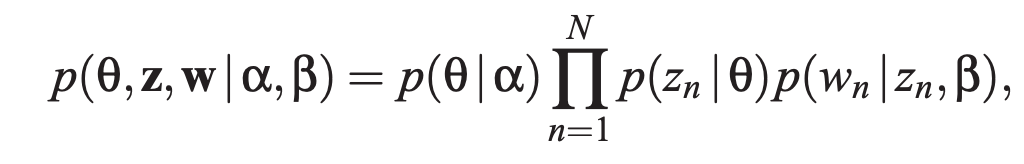
LDA assumes the following generative process for each document **w** in a corpus D:

1. Choose θ ∼ Dir(α).
2. For each of the N words *wn* :

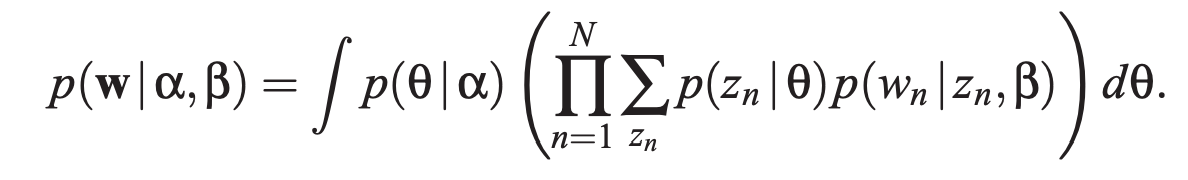
(a) Choose a topic zn ∼ Multinomial(θ)

(b) Choose a word wn from p(wn |zn, β), a multinomial probability conditioned on the topic zn

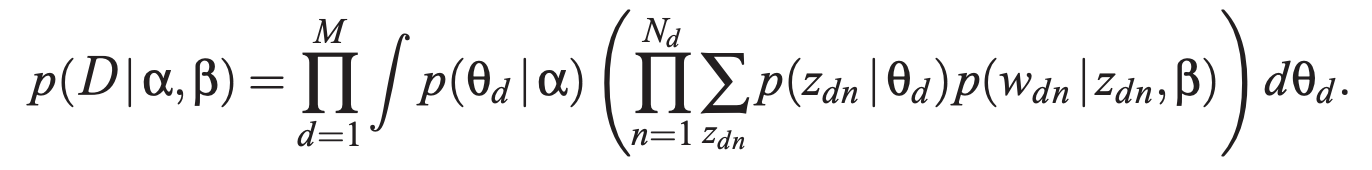
Given the parameters α and β, the joint distribution of a topic mixture θ, a set of N topics z, and a set of N words w is given by:



Integrating over θ and summing over z, we obtain the marginal distribution of a document:

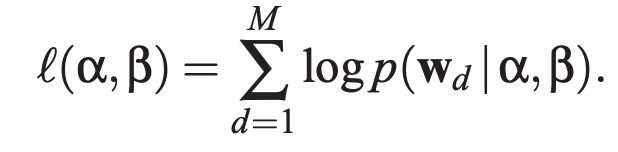


Finally, taking the product of the marginal probabilities of single documents, we obtain the probability of a corpus:

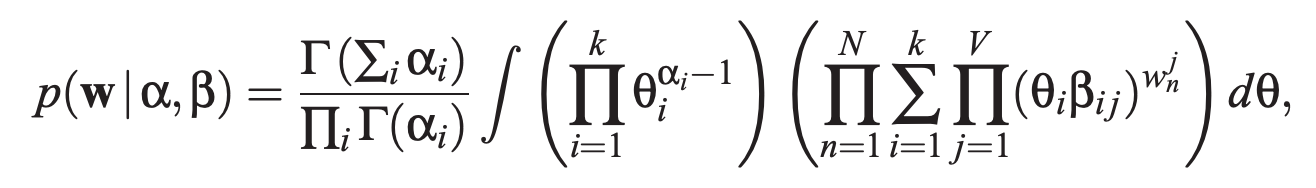


**Parameter estimation**

Given a corpus of documents D = {**w1**, **w2**, ……. **wM**}, we wish to find parameters α and β that maximize the (marginal) log likelihood of the data:



Where



α and β are obtained using optimization steps with the help from following algorithms:

1. Variational inference
2. Kullback-Leibler (KL) divergence
3. EM algorithm

**Python implementation:**

<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.LatentDirichletAllocation.html>

Resources:

<http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf>

<https://towardsdatascience.com/light-on-math-machine-learning-intuitive-guide-to-latent-dirichlet-allocation-437c81220158>

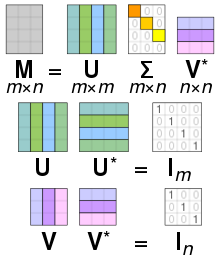
<https://en.wikipedia.org/wiki/Dirichlet_distribution>

<https://www.youtube.com/watch?v=VTweNS8GiWI>

**Other Topic Modelling procedures:**

**Singular Value Decomposition (SVD)**

The SVD algorithm factorizes a matrix into one matrix with **orthogonal columns** and one with **orthogonal rows** (along with a diagonal matrix, which contains the **relative importance** of each factor).



**Non-negative Matrix Factorization (NMF)**

Rather than constraining our factors to be orthogonal, another idea would to constrain them to be non-negative. NMF is a factorization of a non-negative data set V:

*V = WH*

into non-negative matrices W and H. Often positive factors will be **more easily interpretable** (and this is the reason behind NMF's popularity).

