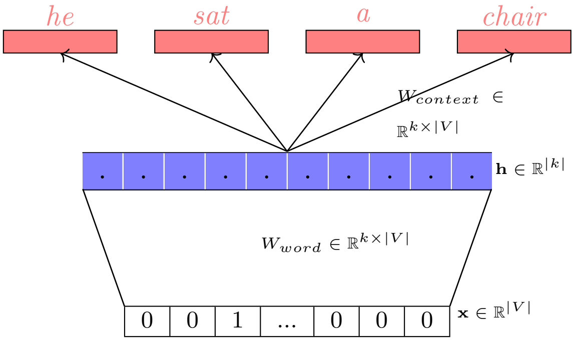
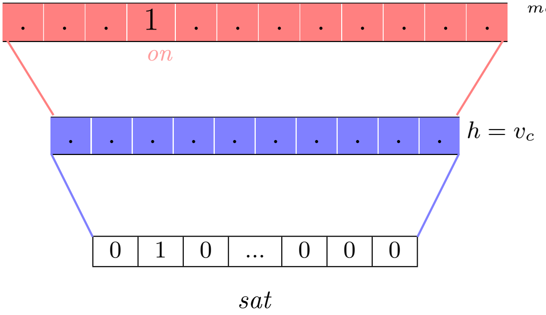
Module 10.7: Hierarchical softmax



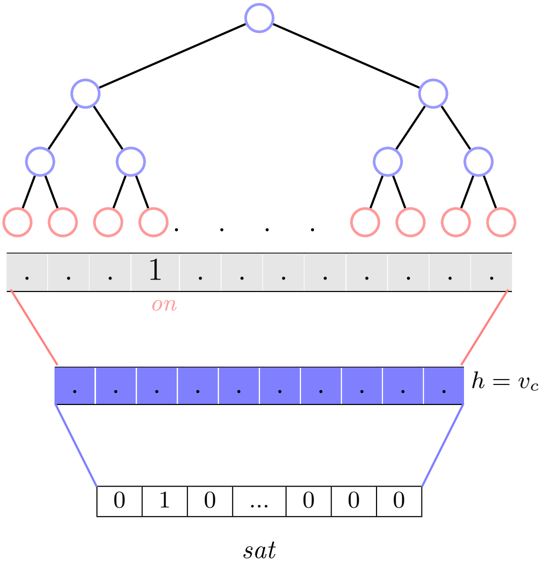
Some problems

- Same as bag of words
- The softmax function at the output is computationally expensive
- Solution 1: Use negative sampling
- Solution 2: Use contrastive estimation
- Solution 3: Use hierarchical softmax

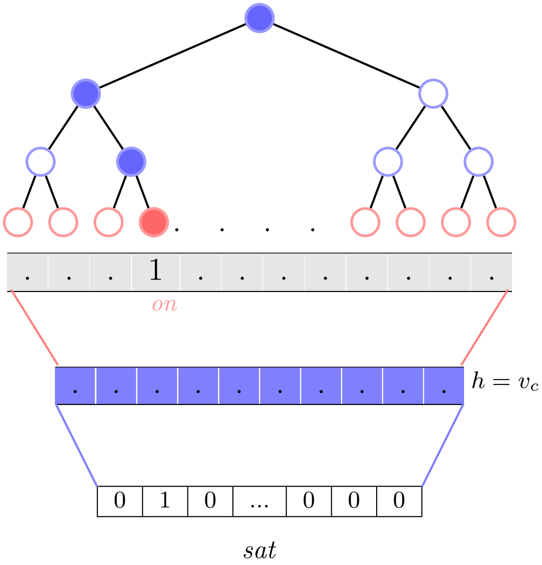
- Construct a binary tree such that there are $|V|$ leaf nodes each corresponding to one word in the vocabulary



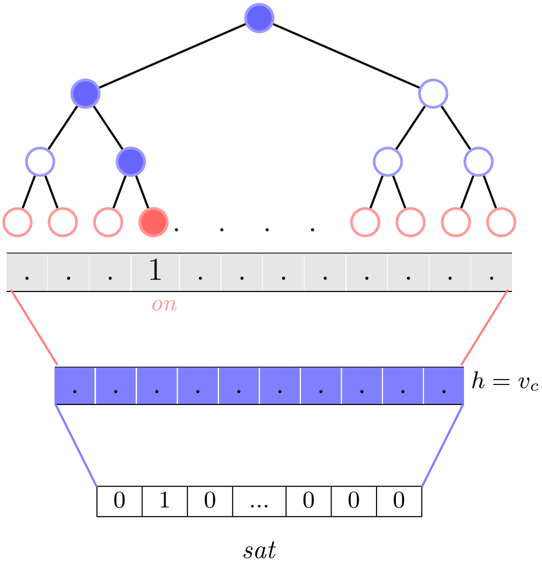
$$max \frac{e^{v_c^T u_w}}{\sum_{|V|} e^{v_c^T u_w}}$$



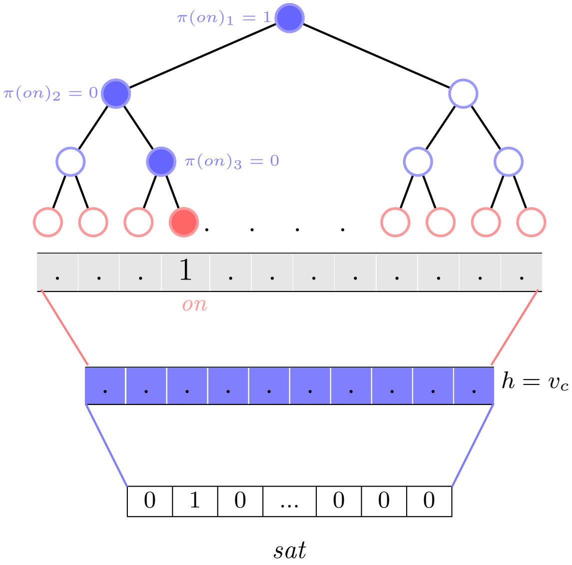
- Construct a binary tree such that there are $|V|$ leaf nodes each corresponding to one word in the vocabulary



- Construct a binary tree such that there are $|V|$ leaf nodes each corresponding to one word in the vocabulary
- There exists a unique path from the root node to a leaf node.

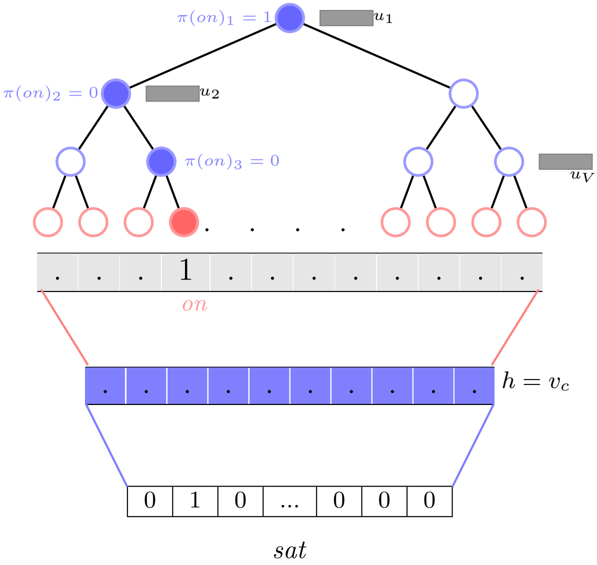


- Construct a binary tree such that there are $|V|$ leaf nodes each corresponding to one word in the vocabulary
- There exists a unique path from the root node to a leaf node.
- Let $l(w_1)$, $l(w_2)$, ..., $l(w_p)$ be the nodes on the path from root to w



- Construct a binary tree such that there are $|V|$ leaf nodes each corresponding to one word in the vocabulary
- There exists a unique path from the root node to a leaf node.
- Let $l(w_1)$, $l(w_2)$, ..., $l(w_p)$ be the nodes on the path from root to w
- Let $\pi(w)$ be a binary vector such that:

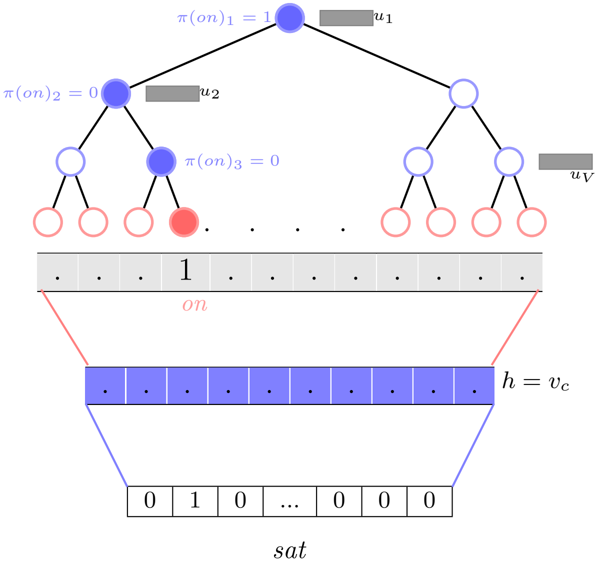
$$\pi(w)_k = 1 \quad \text{path branches left at node } l(w_k)$$
$$= 0 \quad otherwise$$



- Construct a binary tree such that there are $|V|$ leaf nodes each corresponding to one word in the vocabulary
- There exists a unique path from the root node to a leaf node.
- Let $l(w_1)$, $l(w_2)$, ..., $l(w_p)$ be the nodes on the path from root to w
- Let $\pi(w)$ be a binary vector such that:

$$\begin{aligned} \pi(w)_k &= 1 \quad \text{path branches left at node } l(w_k) \\ &= 0 \quad \textit{otherwise} \end{aligned}$$

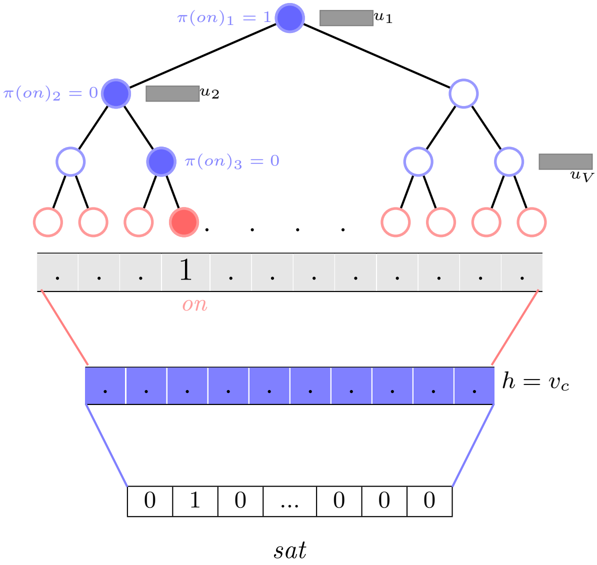
- Finally each internal node is associated with a vector u_i



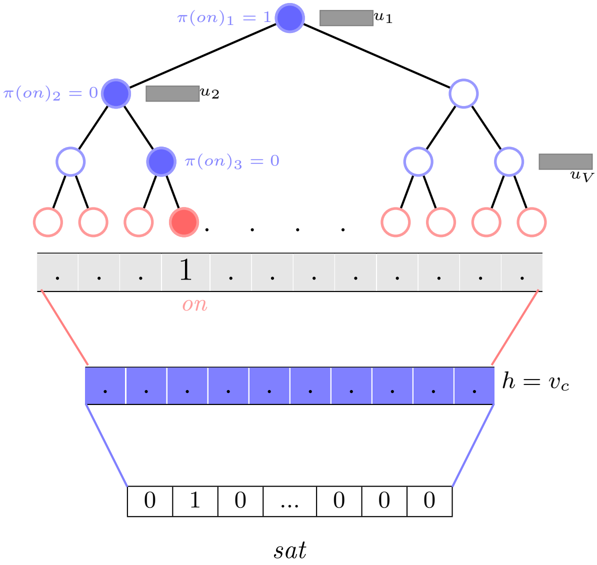
- Construct a binary tree such that there are $|V|$ leaf nodes each corresponding to one word in the vocabulary
- There exists a unique path from the root node to a leaf node.
- Let $l(w_1)$, $l(w_2)$, ..., $l(w_p)$ be the nodes on the path from root to w
- Let $\pi(w)$ be a binary vector such that:

$$\begin{aligned} \pi(w)_k &= 1 \quad \text{path branches left at node } l(w_k) \\ &= 0 \quad \textit{otherwise} \end{aligned}$$

- Finally each internal node is associated with a vector u_i
- So the parameters of the module are $\mathbf{W}_{context}$ and $u_1, u_2, \ldots, u_v$ (in effect, we have the same number of parameters as before)

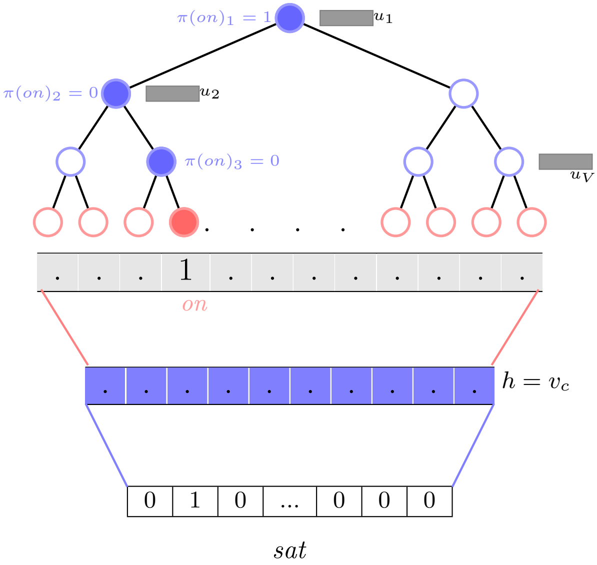


- For a given pair (w, c) we are interested in the probability $p(w|v_c)$



- For a given pair (w, c) we are interested in the probability $p(w|v_c)$
- We model this probability as

$$p(w|v_c) = \prod_k (\pi(w_k)|v_c)$$

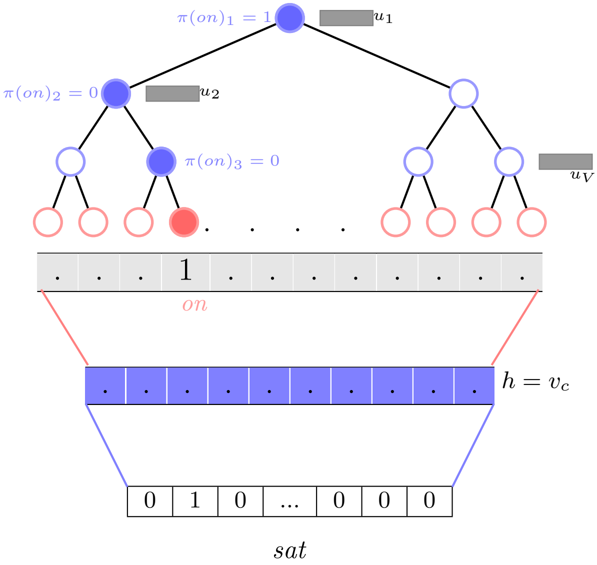


- For a given pair (w, c) we are interested in the probability $p(w|v_c)$
- We model this probability as

$$p(w|v_c) = \prod_k (\pi(w_k)|v_c)$$

- For example

$$\begin{aligned} P(on|v_{sat}) = P(\pi(on)_1 &= 1|v_{sat}) \\ *P(\pi(on)_2 &= 0|v_{sat}) \\ *P(\pi(on)_3 &= 0|v_{sat}) \end{aligned}$$



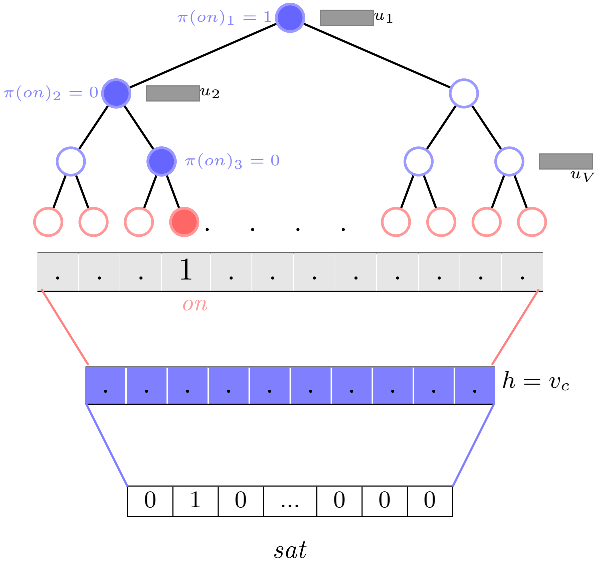
- For a given pair (w, c) we are interested in the probability $p(w|v_c)$
- We model this probability as

$$p(w|v_c) = \prod_k (\pi(w_k)|v_c)$$

- For example

$$\begin{aligned} P(on|v_{sat}) &= P(\pi(on)_1 = 1|v_{sat}) \\ &\quad *P(\pi(on)_2 = 0|v_{sat}) \\ &\quad *P(\pi(on)_3 = 0|v_{sat}) \end{aligned}$$

- In effect, we are saying that the probability of predicting a word is the same as predicting the correct unique path from the root node to that word

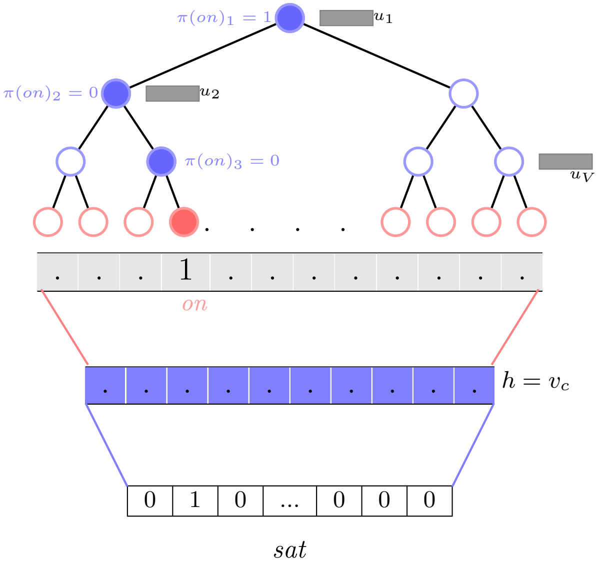


- We model

$$P(\pi(on)_i = 1) = \frac{1}{1 + e^{-v_c^T u_i}}$$

$$P(\pi(on)_i = 0) = 1 - P(\pi(on)_i = 1)$$

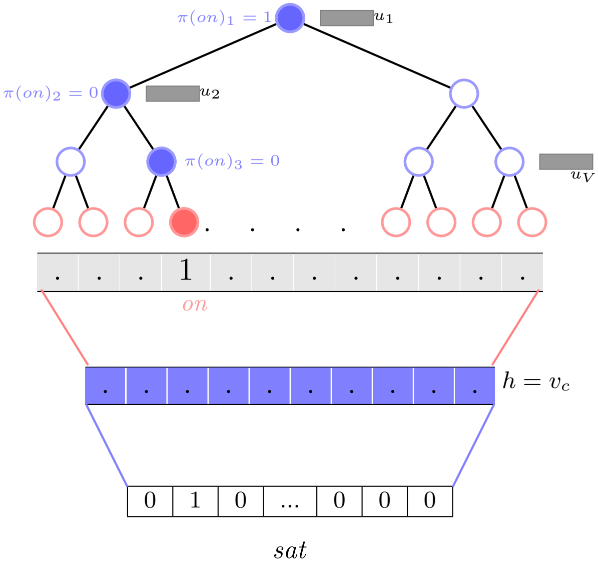
$$P(\pi(on)_i = 0) = \frac{1}{1 + e^{v_c^T u_i}}$$



- We model

$$P(\pi(on)_i = 1) = \frac{1}{1 + e^{-v_c^T u_i}}$$
$$P(\pi(on)_i = 0) = 1 - P(\pi(on)_i = 1)$$
$$P(\pi(on)_i = 0) = \frac{1}{1 + e^{v_c^T u_i}}$$

- The above model ensures that the representation of a context word v_c will have a high(low) similarity with the representation of the node u_i if u_i appears and the path branches to the left(right) at u_i



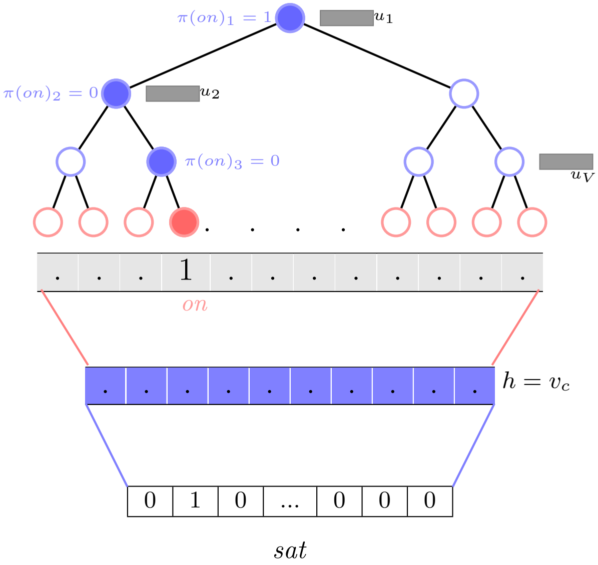
- We model

$$P(\pi(on)_i = 1) = \frac{1}{1 + e^{-v_c^T u_i}}$$

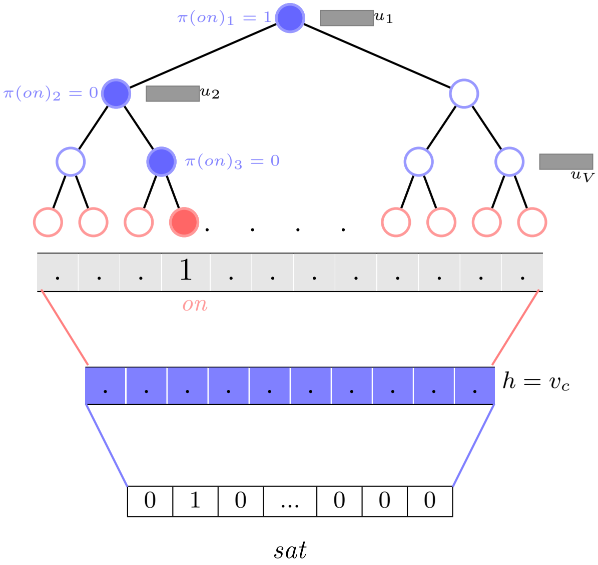
$$P(\pi(on)_i = 0) = 1 - P(\pi(on)_i = 1)$$

$$P(\pi(on)_i = 0) = \frac{1}{1 + e^{v_c^T u_i}}$$

- The above model ensures that the representation of a context word v_c will have a high(low) similarity with the representation of the node u_i if u_i appears and the path branches to the left(right) at u_i
- Again, transitively the representations of contexts which appear with the same words will have high similarity

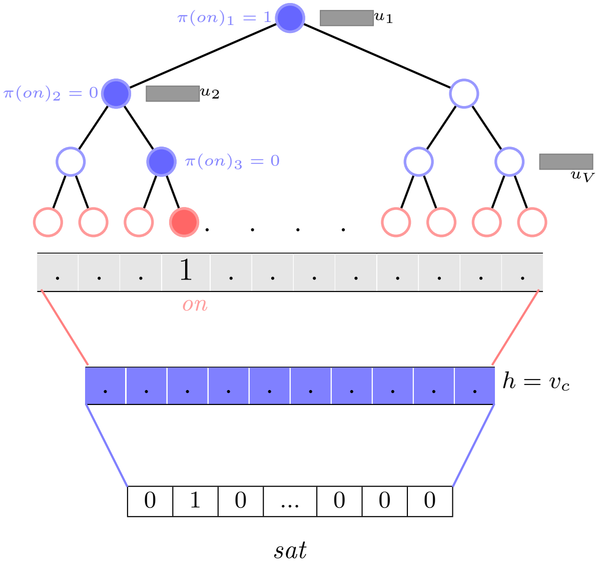


$$P(w|v_c) = \prod_{k=1}^{|\pi(w)|} P(\pi(w_k)|v_c)$$



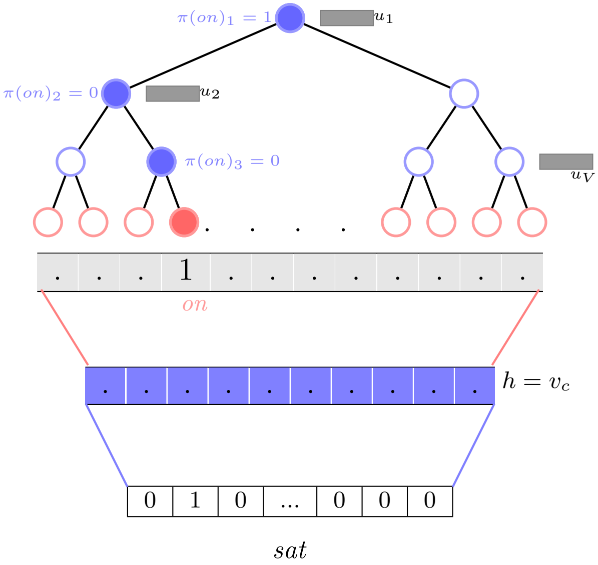
$$P(w|v_c) = \prod_{k=1}^{|\pi(w)|} P(\pi(w_k)|v_c)$$

- Note that $p(w|v_c)$ can now be computed using $|\pi(w)|$ computations instead of $|V|$ required by softmax



$$P(w|v_c) = \prod_{k=1}^{|\pi(w)|} P(\pi(w_k)|v_c)$$

- Note that $p(w|v_c)$ can now be computed using $|\pi(w)|$ computations instead of $|V|$ required by softmax
- How do we construct the binary tree?



$$P(w|v_c) = \prod_{k=1}^{|\pi(w)|} P(\pi(w_k)|v_c)$$

- Note that $p(w|v_c)$ can now be computed using $|\pi(w)|$ computations instead of $|V|$ required by softmax
- How do we construct the binary tree?
- Turns out that even a random arrangement of the words on leaf nodes does well in practice