n-gram modeling has a fundamental limitation

- In lecture-03 we learned that word2vec was motivated by this idea of capturing the meaning of words within their contexts. We didn't formalize the learning problem as one of language modeling, but in fact it is a language model that uses the skip-gram assumptions (Markov assumption + context word ordering doesn't matter).
- But, are we really solving the problem? We've learned that *n* is practically limited to around 10 or so; even at n=10 we have a sparsity problem and are forced to introduce bias in order to even compute the probability of sequences. We can hardly say that we're capturing context needed for understanding. For example, any reasonable model of human language should be able to capture the intended meaning of "*crashed*" in the following sentence:

Examples from Eisenstein (2018)

The **computer** that's on the 3rd floor of our office building **crashed**.

- In practice, computing *n*-gram probabilities from occurrence frequency is adequate for some tasks, and far from adequate for others.

Character *n*-grams

- Instead of treating each word as unique, why not treat sub-words as unique?
  - Words share characters, and more generally, n-grams of characters!

- Words get factored into (potentially overlapping) character n-grams.
  - Example: *artificial* = <ar, art, rti, tif, ifi, fic, ici, ial, al>

- Allows us to share features between words, as character n-grams are shared across a large number of words.
  - Example: *ben, ten, happen, entropy, envisage, envy, envoy, been, seen, lend*

- Inspired by Word2Vec with similar training procedures

- Computational efficient relative to deep learning based methods, while offering very competitive performance on classification and retrieval based tasks.