EPISODE 48

[INTRODUCTION]

[0:00:10.5] SC: Hello and welcome to another episode of TWIML talk. The podcast where I interview interesting people doing interesting things in machine learning and artificial intelligence. I'm your host Sam Charrington. A big thanks to everyone who joined us last week for our second TWIML online meetup led by Nicola Kucherova, we discuss the paper, *Learning Long Term Dependencies With Gradient Descent Is Difficult* by Yashu Abengo and company.

I had a great time and learned a ton. For those who weren't able to attend live, we have the video posted for you on twimlai.com/meetup and if you're interested in joining us next month, please head on over to that site to get signed up.

We're also accepting presenters so feel free to shoot me a note with your ideas. Next up, the time has come for the Artificial Intelligence Conference brought to you by O'Reilly and Intel Nervana. I'll be in San Francisco the entire week next week and we have a ton of great interviews lined up.

This should be another awesome series. If you haven't had a chance to check out our series from the New York event, you can find it at twimlai.com/oreillyny. Also, if you'll be at the conference, please send me a shout out on Twitter or via email. I'd love to connect. See you then.

Okay, about our show.

This week, I'm bringing you an interview with Bruno Goncalves. A Moore Sloan data science fellow at NYU. As you'll hear in the interview, Bruna is a long-time listener of the podcast. We were able to connect at the NYAI conference back in June after I noted on a previous show that I was interested in learning more about word2vec. Bruno graciously agreed to come on the show and walk us through an overview of word embeddings, word2vec and a bunch of related ideas.

He provided a great overview of not only word2vec but related natural language processing concepts such as Skip Gram, Continuous Bag Of Words, note2vec, TFIDF and much more. By the time you hear this, it will be too late to catch it live but Bruno is doing a half day tutorial on word2vec and friends on Monday at the AI conference.

If you'd like to see it, I'm sure it will be available via the O'Reilly Safari website. Now, on to the show.

[INTERVIEW]

[0:02:49.5] SC: Alright everyone, I am here on location at the O'Reilly AI conference and I have the pleasure of being here with Bruno Goncalves who is apparently a long time listener of the podcast and reached out to me after hearing me note that I wanted to learn more about word2vec and embedding and made it happen. We're here to talk about that. Welcome Bruno.

[0:03:12.5] BG: Thank you, it's a pleasure to be here as I've been listening for a very long time and I'm very happy to be here and be able to participate.

[0:03:18.8] SC: Awesome. I was just asking you, you mentioned that you've been listening since before the interview format and I don't know, I'm still stuck on the transition from the news format to the interview format. How was that for you? What's your take on kind of the before and after?

[0:03:35.4] BG: I like the new interview format actually. This might be a personal bias. When it was news, a lot of the news I'd already seen or listened somewhere else so it wasn't as new. This way, since you're interviewing people that I don't necessarily listen to and not know personally.

It's good to have a fresh view from them.

[0:03:53.8] SC: That's great. Like I said, I was, I guess, I still am hung up on the fact that I switched the format but I've been told by people that things along the same lines that the news

is essentially commoditized and it's hard to beat out Tech Crunch or whatever that people count on for their news.

The folks really seem to be enjoying the interview format and so you know, again, it's great to have you on for an interview instead of talking about the news. Tell us a little bit about kind of your background and what you're up to?

[0:04:25.3] BG: Like I was saying, I'm originally a physicist but I've always been involved in optimization problems, originally in spin glasses, optimization and data mining.

[0:04:32.8] SC: Spin glasses?

[0:04:34.9] BG: It actually has this interesting connection with neuro networks and how they work but basically a disorder magnetic system.

[0:04:40.7] SC: Okay.

[0:04:41.6] BG: To put it very simply. But, the mathematical structure is very rich and it leads itself to exploring different types of optimization problems. From there I evolved very quickly and because I noticed that in spin glasses, a lot of the behavior you see has to do with the way that the different elements are connected.

I moved very quickly towards networks and studying basically connections between components of a system more deeply. When you're working with network, the complex network in general, what you do a lot is basically data mining.

You're crawling websites, you're parsing up entry logs to look for connections between URL's for example. You're basically doing in a sense, applied graph theory but applied to real data.

The transition towards data became very early and very naturally, even before, I mean, people were talking about data science, this was back in maybe 2006, 2007 and then more recently, I've been moving more towards more data science.

Intensive aspects and hence my interest in word2vec and that type of algorithms.

[0:04:41.6] SC: Okay, you know, I think the best way to do this is to just jump in and have you kind of walk us through word2vec but kind of the foundational thinking that led to word2vec and embeddings. As I mentioned the other day, it's an area that I've been meaning to dive in to and I'm glad you're here to tell us about it.

[0:06:07.8] BG: Excellent. So, the original idea is actually as many things doing these field comes from outside the field. So it's come from linguistics and it's expressed more or less in a very general way by James Firth if I'm not mistaken, around 1953.

Basically what he said is, you know the meaning of the word by the company it keeps. If you don't understand what the word means, you kind of look at the context and the context gives clues about what the word says, what the intent is to say.

Word2vec tries to take advantage of this by saying that the embedding of a word is defined by the context in which it appears. Words that appear in the same context are more somewhat equivalent. So they will have vectors that are close to each other – words that do appear in very different contexts will be very different and have very different vectors. Or vector presentations in this case.

[0:07:05.0] SC: How do you get to the word vectors? I guess, my impression is that it's all relative to a specific corpus, right? There is not like – we haven't done word2vec on everything like some grand unification of word2vec.

Is that the right way to think about it? And then you're doing some math on that corpus to get you to the vectors, can you talk a little bit about that process?

[0:07:27.6] BG: Yes, basically, what it does, it's a very simple neuro network. Just one hidden layer, the activation function is linear so it just passes through, so it's very simple. What the actual word embeddings are basically the weight on this layer.

Depending on which model, it can be Skip Grammar, it can be Continuous Bag Of Words, it can be the weight on the leading to the hidden layer, it can be the weights outside heading out of the hidden layer but in practice, it's the same mathematically.

[0:07:59.1] SC: The Skip Gram refer to one of those and Continuous Bag Of Words refer to that, the skip gram is leading in, the waste leaning in and…

[0:08:06.7] BG: Yeah, the way you look at what I was saying that you know a word by the company it keeps. If you have the word, you can kind of guess the context in which it appears or if you have the context, guess what word would fit in the middle.

The Skip Gram and the Continuous Bag Of Words basically look at these two approaches. On one case, your input is the word and you're trying to guess in the output layer what the context is. On the other case, you have the context where is this input and you're trying to guess what is the word that might go in the middle.

[0:08:37.3] SC: Okay.

[0:08:37.9] BG: Just to clarify, the context is usually defined, your window of worlds before that word and the words after that word. If you're looking at word I, your context would be for instance, I minus 1. I minus two and I plus one, I plus two. The words before and towards after of five words before five words after.

[0:08:56.5] SC: Okay. That window is what you use to compute the vector for a given representation for a given word.

[0:09:04.6] BG: Yes, that window will give you what is the context and the context will define what is the word.

[0:09:09.1] SC: Okay.

[0:09:10.0] BG: Now, like you're saying, people haven't don't word2vec in all the text in the world. Every time you run this, you will get different vectors.

[0:09:17.6] SC: Alright.

[0:09:18.1] BG: Also, if you run twice in the same corpus, the vectors will be different. The reason for this is, you're initializing all the weights, all the vectors randomly and then you're adjusting them so you wind up with something different. However, if you do it well enough and it converges and you get something this reasonable, the vectors will be different but there will basically be a rotation. You can always get vectors trained into corpuses and align them so that they match.

What the word2vec algorithm and see the algorithms are trying to do is learn the relations between vectors. So you're basically, you're trying to define the differences of vectors but not the actual vectors, right?

[0:09:57.0] SC: There's some distance metric or something here.

[0:09:59.1] BG: If you rotate, the distances all remain the same so there's two on them.

[0:10:02.3] SC: Okay.

[0:10:03.0] BG: It's because the distances are preserved, it's the distance that preserves the semantic meaning. That gives you the semantic relations, right? The idea is very simple. If the same word, if two words appear in the same context, they have to be defined by similar vectors.

This also means that if the relations between this pair of words and relation between this pair of words are similar, the difference between them in terms of vector will also be similar.

[0:10:29.4] SC: Okay.

[0:10:30.1] BG: This is why you can do word arithmetic and say, the vector for Italy minus the vector for Rome has to be equal to the vector for France, minus the vector for Paris.

[0:10:40.4] SC: Right.

[0:10:41.2] BG: If you have three of these, you can calculate the other one.

[0:10:43.3] SC: Right.

[0:10:43.3] BG: Basically it's one of the ways they used to calculate or to measure how good the embeddings are, they just called it analogies, right? Paris is to France is Rome is to give you Italy. If you look for the vector, that's closest to that difference.

[0:10:58.6] SC: Okay, interesting. One of the questions that – well, two questions come out for me. One is, if the context is defined by this narrow window, you said it's usually used two words before, two words after.

[0:11:12.5] BG: I give the example of two words and in the official implementation, I think the default is set to five but it depends, you can vary it. It's an input parameter and also in practice, if you want to go into the details, when you're preprocessing the corpus, you will sometimes remove some words because they're too common. Kind of like with stop words and that effectively changes the size of the window because you do this before you calculate the context.

It's almost as if sometimes your windows are a bit bigger. Because remove the word so you're catching more information.

[0:11:46.1] SC: Okay, the question is, do folks experiment with bigger windows or is it possible to do this on the entire corpus and get – I guess in my mind what I'm hearing is the word relationships are only relative to this very small window and wouldn't we have – wouldn't vectors capture more information if we were somehow creating them based on bigger windows or the entire corpus, is that the right way to think about it?

[0:12:11.9] BG: Yes and no. If you make the window too big, you're basically including information that's not relevant for the word. You can have a very long sentence, the word at the end of the sentence doesn't necessarily have anything to do with the word at the beginning of the sentence, right?

You're trying to keep nearby words that help you define what that word means. Adjectives, verbs, the directory related to what the concept is, what the word is in particular.

[0:12:38.0] SC: Right. I guess I'm thinking about it in the context of like running TFIDF on a Wikipedia article, right? If I'm looking at a Wikipedia article that's talking about neural networks and CNN's and RNN's and you know, artificial neuro networks, all these – these are all related terms and I'd want to capture that relatedness but they may be in very different…

They may be far from one another spatially in my document but I still want to capture that context. Can I just not use word2vec for that?

[0:13:16.0] BG: You can. Basically what you're doing is you're scanning through the entire document and embedding that you learn for each word. Has to do with all the context it appears in. It's not just one.

I could give the example of artificial neuro network, right? Artificial will appear next to neuron in this context. Maybe a few paragraphs later, it will appear next to intelligence or to approach, right? That means that artificial will be defined by all of this context and look very different than maybe like evolution.

That always appears next to neuro network and doesn't appear in other context. It does take the whole information of the corpus into account.

[0:13:59.2] SC: If for example I'm running it on a Wikipedia article on neuro networks and there's one section on convolution on our network and then another section on RNN's and another section on LSTM's. Since those individual terms are separated by these sections.

Will it capture those relationships via just the neuro network part or…?

[0:14:21.3] BG: Yes, so one thing, maybe I should include. When I say corpus in this case, I mean, all of Wikipedia. I don't mean one page of Wikipedia.

[0:14:28.4] SC: Okay.

[0:14:28.8] BG: These are very large corpuses.

[0:14:31.6] SC: Got it.

[0:14:32.1] BG: The reason why you use a very large corpus is because you want to learn what is the meaning of the word in a sense. This is what you're trying to capture. It's not necessarily what does this word mean in this document.

[0:14:44.7] SC: Got it.

[0:14:44.8] BG: It's more of – it's a more generic thing.

[0:14:47.3] SC: Okay.

[0:14:47.8] BG: This is why actually people have started publishing high quality embeddings. Google has published some, Facebook has published. That are trained on billions of words or rother corpora that are billions of words long because they're trying to capture what is that meaning.

You can use, for instance, these vectors, one very simple application is, for example, for quarterlies ambiguation. Because the word, you know, what a vector is, you can look around that word to see what are words that are related to that one and maybe you can show results that use those other words.

You can use the vectors and this translation in various distance, preservation as a way of mapping the word, for instance to a verb version of the word or a noun version of the word or a past tense version. You can use all of these relations in a sense, all of these linear algebras has a way of getting more knowledge about what the text is.

[0:15:38.3] SC: Okay, interesting. The other question I had was the length of the vector. How do you know what the right dimensionality is for these vectors?

[0:15:49.6] BG: As far as I know, there is no well-defined way of measuring what is the optimal size.

[0:15:54.6] SC: Okay.

[0:15:55.0] BG: In practice, people tend to use dimensions between about a hundred and 500.

[0:16:01.2] SC: Okay.

[0:16:01.8] BG: Which is relatively small for corpora or rather, dictionary. The number of individual words of the order of hundreds of thousands. In a sense, one of the things you're also doing is you're doing very much a dimensionality reduction. Mapping this from this very large high dimensional space where each dimension is a word, into this very small space. In a way that is still preserving the meaning of the semantic value of the words.

[0:16:27.9] SC: Okay. Do you recall the show that I did with Francisco Webber?

[0:16:32.4] BG: Maybe. It was from Corticol that I own, he talked about the kind of neuro representations. Yes, I actually remember that I wanted to look into that more carefully. Because it exquisitely mentioned that this is actually related with this type of representations because it seems that each word is actually represented in different parts of the brain at the same time.

[0:16:52.5] SC: Right.

[0:16:53.0] BG: It's something I'm curious to look more deeply into it. I tried looking at their website but I couldn't get the technical details.

[0:17:00.8] SC: Okay, I was curious if you had looked into that at all and if you are familiar with that model and any thoughts you might have on how it compares to…

[0:17:09.1] BG: I found some things online but it seems to be more marketing oriented so there wasn't like the scientific articles we had.

[0:17:14.6] SC: Okay. Embeddings has been around since the 50's, word2vec now is – didn't you say 1950's something?

[0:17:22.2] BG: 1950's is the idea, is this idea that the meaning of the word is coming from the context it appears. This is called the distribution of the hypothesis in linguistics.

[0:17:30.9] SC: Okay. Embeddings themselves?

[0:17:32.5] BG: Embeddings have been around for maybe 10 years, maybe a little bit more. Okay, in practice, I can wind the option. They became very popular and get a lot of attention with word2vec, when word2vec was published in 2013 by Tomas Micolov if I'm not mispronouncing his name horribly.

[0:17:49.3] SC: Okay. You know, we're a few years into word2vec and now there are a bunch of other 2vecs, right? Have you looked into any of those as well?

[0:17:59.5] BG: Yes, so there are some that are very interesting. There's one for instance that is DNA2vec and he tries to find embedding's for sequences of DNA and see how that might relate to protein structure and genome organization, which I find fascinating. There is one by Jure Leskovec in Stanford where it's called node2vec and it tries to use this –

[0:18:23.3] SC: Note or node?

[0:18:24.7] BG: Node, like in a graph, where it's basically trying to do that to find embedding's for nodes and graphs as a way of measuring relations between nodes. Basically the way it does it since he doesn't have a sequence of words and it doesn't have a sequence of nodes. So he basically runs a random walk process on the net on the node and that generates the sequence and then based on that sequence then you can treat each node as if it was a word.

It appears next to other nodes because they're somehow in the neighborhood and from there you can define and from that it is able to capture some of the structure of the network which is kind of interesting.

[0:18:58.5] SC: So the idea is that you're – I guess I would have thought that a graph is its own kind of representation of all of this information.

[0:19:07.5] BG: It is and it isn't in a sense right? The point is it's not necessarily and this is a one on problem to compare graphs, right? So I'll give you a graph and tell you the nodes here are words and then I would give you another graph that then we did nodes here are people. So you can just match the labels directly. It's very hard to see if the structure of the graph is the same because you actually get all permutations with all the things.

So if I am not mistaken I think it's like an NP complete problem. So people have been trying this idea of graph embedding's for a while, where they try to map the graph into some set of points that does not depend on the details of the graph or the labels that you give to nodes or anything like that or any permutations you do.

[0:19:50.3] SC: Interesting.

[0:19:52.2] BG: So I have been fascinated by basically how versatile this idea of word2vec is.

[0:19:57.1] SC: Yeah, are there others? I'm at a loss for – I know I have seen it. It seems like a dozen of these different 2vecs of late. But it sounds like anytime you have a space with some complex structure, this is a tool that you can use to allow you to compare both individual points within that space to other spaces. Is that a general way to think about it?

[0:20:22.8] BG: It's more whenever you have a sequence of tokens let's say. You can use it as a way of finding a representation of these tokens, that then you can use to find the relationships in this sequence, right? So it works very well for text because that sequence is of words. Like in the case of dna2vec, it works very well because you have a sequence of nucleotides in DNA. For node2vec, you have a sequence of nodes that are generated by these random processes.

These random walk processes on the graph and they actually tried different definitions of the random walk, different rules. So that somehow has been able to capture different aspects of those nodes of the network structure.

[0:21:00.6] SC: Interesting. Then do the sequences have to have – I guess the relationships between the tokens and the sequences are defined by the corpus? I guess I was thinking of I wonder if you can do like transaction2vec and do word and embedding on the sequence of transactions and use that for a fraud detection or something like that.

[0:21:22.3] BG: I actually saw something like this yesterday. I didn't look because I didn't have time but I don't remember it but it was really something like stock2vec that somebody posted on Medium yesterday. It's on my reading list but I have not read it yet.

[0:21:37.3] SC: Alright, interesting.

[0:21:38.5] BG: So I don't know how they are doing it. I just saw the title.

[0:21:41.1] SC: But notionally there's something in there somewhere.

[0:21:44.2] BG: Yes, so in principle every time you have a sequence you can do something like this.

[0:21:49.0] SC: I have also been getting more into learning about Block Chain and how that all works in this and I am wondering now if there are some applications to that as well.

[0:21:58.0] BG: Possibly. I have not seen anything with Block Chain using this. I have looked in detail of Block Chain a couple of years ago because I was interested in basically these transaction networks in a sense. It might be possible to do something. I said I haven't seen it yet – maybe an idea for some of your listeners.

[0:22:15.9] SC: Yeah, what are some of the most interesting applications you've seen of word2vec and friends?

[0:22:25.7] BG: I was fascinated by basically how much power these word vectors have because they are capturing these semantic relations. It means that you can use them for design ambiguation, quarry expansion analogies – like the original metric that they used. So basically they are very powerful tool to map text into a vector representation or to an American

representation, that then is very powerful in how you can manipulate it. Because everybody knows computers have a hard time understanding text.

But they are very good at understanding vectors, so here you are mapping text to vectors, so you are making computer's life much easier. So people have used this for machine translation because you can translate vectors into different talk with vector embeddings for two different languages. Then you align them that's because the relationships have to be the same mostly and then you can use them basically to match. So you find the word embedding in one language.

You find what is the most similar word, most similar vector in the upper language and you can find the translation.

[0:23:25.6] SC: Also an idea that came up in the Francisco Webber conversation. So yeah, we both need to dig into that one, try to figure out what they were doing that's different than regular embeddings.

[0:23:36.5] BG: I mean there are other variations, there is also Glove, global factors that tries to do a different formulation would be recommended. Another very interesting application, which is actually the reason why I started getting interested in word2vec in particular, is this paper I saw which is actually tracking linguistic change overtime. So they train is in word2vec, they train word vectors using Google anagrams data for different decades. Then they align the vectors and then they look for the words that are changing overtime.

[0:24:08.7] SC: Oh that's interesting.

[0:24:10.2] BG: But you can track basically how the meaning of the world is evolving.

[0:24:13.6] SC: Is there a metric for measuring the, I guess like dispersion of the vectors in a given embedding. Like I'm wondering in that example where you train an embedding on this engram data. Can you look at something analogist to like the standard deviation of the words or like the spread or the degree to which the –

[0:24:39.2] BG: Not really, I don't think no. I mean if you look at the entire set of vectors by itself. I don't think so because what you're doing is you are starting with a bunch of random vectors and then you are slightly adjusting them so that specific pairs have specific relations between them, right? And that specific groups have specific relations between them so they are more or less the same distance from each other.

But all of these groups and all of these words can be arbitrarily rotated with respect to each other. I suspect that if you just get even very high quality embeddings and you just start measuring distances between them, it will look random because you are putting in too much noise. The distance between, I don't know, bed and blue doesn't mean anything.

[0:25:23.4] SC: It's kind of immaterial right. But then more generally there's not a set of statistics or metrics or things that are relevant at the aggregate level for the embedding?

[0:25:36.5] BG: Not that I know of, no. Okay, usually what people do is they look for a specific – relations between specific words so they do this analogy problem. That's how you check that what you are capturing is what you think it is. To actually capture the semantic meaning of the words and these semantic relations between words.

[0:25:56.5] SC: Why do we end up using neuro nets to do embeddings? It seems like a pretty basic math that you can do more deterministically?

[0:26:05.9] BG: Yes. So I think that is one of the motivations for people to do, to invest in Glove and Glove the abbreviation for Global Vectors. This is, I want to say it is coming from Stanford. I might be wrong. Basically they try to do it. Basically in the deterministic and defining optimization process by specifically saying, I want vectors that where the relationships are given in this specific way. The reason I think one of the motivations for using neuro networks for this is because now you have very powerful tools to optimize and train neuro networks in large scales.

So you can use all of that machinery because when you actually look at the mathematics and the network structure it's actually using, it is actually very simple. So it's an extremely simple neuro network. Where it's mostly vector multiplications and then a soft max at the end with

some exponentials. So it's nothing particularly sophisticated. If you look at networks like Image Night to Alex Night or something that has dozens of layers and are very complex.

So you have all the technology to develop for these very complex things that is being applied to something that is very simple, so it makes it very efficient.

[0:27:17.7] SC: So single layer, the matrix multiplications are just applying your weight coming in and coming out. Then remind us what Soft Max is?

[0:27:24.6] BG: Soft Max is just an optimized way basically to calculate what is the maximum value of the vector, in a very simplified way. While at the same time basically turning a vector of numbers into something that is normalized. So the only thing you do is for each element of the vector just take the exponential of that value and then you divide by the sum of all the exponentials. That's it, that's Soft Max. What that means is that it basically makes any value that is likely larger than the others, will become much larger.

So it's easier to capture the maximum value and then of course there is many optimizations on top of that. That's the general idea because you don't necessary want to calculate all the exponentials for the entire vector. Because these vectors are basically one hot embeddings of the words in the corpus. So this could be a 100,000 or a million words. So you don't necessarily want to have to do that at every iteration. So then there's hierarchical Soft Max, there's all sorts of tricks to train and optimize this.

[0:28:21.8] SC: More like computational tricks for…?

[0:28:24.0] BG: Yes but the concept is very simple and then then there's of course optimizations that people apply to it to make it more efficient and more robust.

[0:28:30.9] SC: Okay and so what's the relationship between one hot encoded 10,000 dimensionality vectors and the hundred to 500 vectors?

[0:28:43.1] BG: So like I was saying before the computers have a hard time understanding text but they are very good at understanding numbers. So what you do is you map in the beginning

the first approximation, is you map words to numbers. So just have a dictionary, this is what number one, this is what number twos.

[0:28:57.3] SC: That's one hot encoding.

[0:28:58.9] BG: Yes or no, your one hot encoding. So it's just the way you represent the words so then you can manipulate them numerically, so that you can calculate these vectors.

[0:29:07.3] SC: So I'm imagining that's your very first step then your input layers and your – well it's just one layer. But your inputs, you are sending in the inputs to the network is the one hot encoded corpus.

[0:29:22.4] BG: So basically you have the input values here and these are let's say in the instance of Skip Gram, this is just the context form that you have in them. This would be if your dictionary is 10,000 words, this would be one hot encoded 10 dimensional vector. This gets fed into the hidden layer in the center which is the dimension of the embedding. It should be say 300 and then from this hidden layer, you are trying to calculate the context and the context can be depending on the window size.

It can be let's say 10 times the size of the dictionary. So it would be 10 times 10,000 so it is a 100,000 because they are trying to predict 10 words, five words before, five words after.

[0:30:09.1] SC: Okay, so then you got your dimensionality 10,000 or so vector coming in. Your network is the dimensionality of your hidden vector, that's your hidden layer and then the output side is a 100,000 you were saying? Like 10 times 10,000?

[0:30:27.5] BG: Yeah, if your window size is five, you're trying to predict all the context from the single word then you are trying to go from 10,000 to 300 to 100,000.

[0:30:37.1] SC: But you are not really using the 100,000. You are just using that as kind of an optimization to get at the weights for the 300 and that's what you use?

[0:30:45.5] BG: Yes, so basically you can think of word2vec actually as unsupervised learning problem because you are feeding it the inputs and the outputs. So that you can learn something about the system. So you are not really – in practice, you have never really trying to predict the output. In a sense I guess you could use that if you are trying to generate that. Use it to generate text without anything unusual. So you are trying to basically see what the network is learning from the text that you are giving it and you are giving it a word and a context and you say:

"Okay figure this out. Figure out what is the right representation that from this input can generate this output" Then in the end, what you are interested in is actually this internal representation. This word embeddings, these vectors.

[0:31:28.6] SC: Awesome. This has been super helpful. I feel like I finally understand them now.

[0:31:33.1] BG: It's easier to explain with pictures and drawings.

[0:31:34.6] SC: For folks that are not in the room, you know he's like moving his hands around and like drawing, you know lays in the air and stuff like that.

[0:31:44.1] BG: I can send you the link to my slides if you want actually?

[0:31:46.2] SC: Yeah, I was actually just going to ask. If someone wants to learn more, what is the best way for them to learn more?

[0:31:50.5] BG: I mean all of these papers are online. You can find them easily. I am preparing this tutorial for O'Reilly in San Francisco in September. I will post all the slides and all the code in my GitHub. Right now, there is already the slides for a shortened version that I presented a couple of weeks ago. So I can share that with you and then eventually after O'Reilly, I will update it with all the newest stuff.

[0:32:13.9] SC: Okay, great. Yeah so send that over and we'll get that in the show notes and yeah, great. Thanks so much for stopping by.

[0:32:19.9] BG: Thank you for inviting me. It was a pleasure.

[0:32:21.7] SC: Awesome.

[END OF INTERVIEW]

[0:32:26.6] SC: Alright everyone. That is our show for today. Thank you so much for listening and of course, for your ongoing feedback and support for this episode head on over to twimlmi.com/talk/48. If you like this episode or have been a listener for a while and haven't done so yet, please, please, please take a moment to jump on over to iTunes and leave us a five star review. We love reading this and it lets other know that the podcast is worth tuning into. One last note, you've probably heard me mention Strange Loop a great technical conference held each year right here in St. Louis. We are a bit over a week away from that conference so I encourage you to check them out. It's thestrangeloop.com, I'll be there and let me know if you'd be there too. For more info on any of these events, check out the show notes. Thanks again for listening and catch you next time.

[END]