Blog Source File

Prashant Tak

April 10, 2021

Contents

1 TODO

1. Fix Time
2. Add Sections
3. Link to RSS feed
4. Add dark mode toggle
5. ~~Look into whether latex can be used and generated properly for displaying notes~~ Improve mathjax performance
6. Create Tufte Theme for hugo that ACTUALLY works based on tufte css and the current tale theme
7. ~~Replace with material scroll bar~~
8. Open external links in new tab by default, unless it's meta-links.
9. Improve Image filter sigh

2 Doubts

1. What is r?
2. dev-corpus refers to?
3. F1 score? Measure of accuracy
4. xpos?

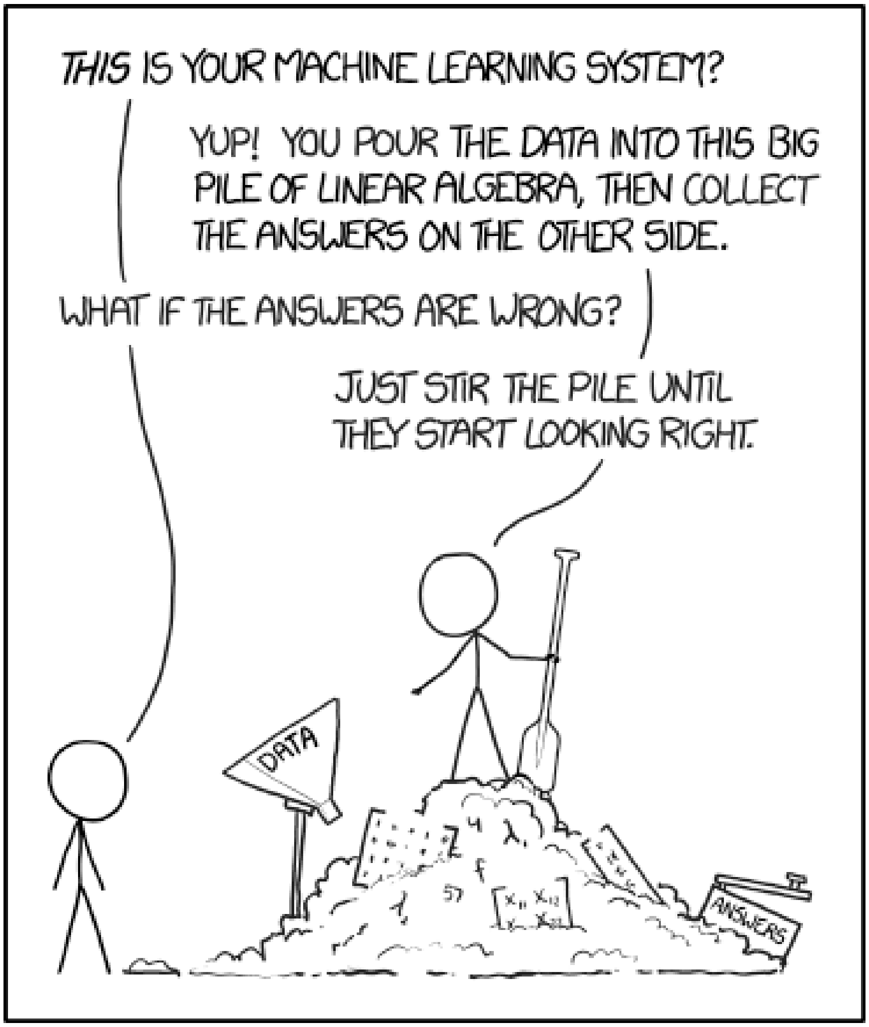
3 About

Hi! I'm Prashant.

4 Blog

4.1 Morphosyntactic Tagging with a Meta-BiLSTM Model - An Overview

(Subtitle: *I had shingles, which is a painful disease.*)


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

This post contains a complete overview of the titled paper and provides a basic outline of related concepts. This paper aims to investigate to what extent having initial sub-word and word context insensitive representations affect performance.

4.1.1 Abstract

1. RNN leads to advances in speech tagging accuracy Zeman et al

2. Common thing among models, *rich initial word encodings.*

3. Encodings are composed of recurrent character-based representation with learned and pre-trained word embeddings[1].
4. Problem with the encodings, context restriced to a single word hence only via subsequent recurrent layers the word information is processed.
5. The paper deals with models that use RNN with sentence-level context.
6. This provides results via synchronized training with a meta-model that learns to combine their states.
7. Results are provided on part-of-speech and morphological tagging[2] with great performance on a number of languages.

4.1.2 Terms

1. Morphosyntactic = Morphology + Syntax and Morphology is study of words, how they are formed, and their relationship to other words in the same language.
2. RNN: On difficulty of training RNNs
3. LSTM: Long Short-Term Memory is a type of RNN that addresses the vanishing gradient problem through additional cells, input and output gates.
4. BiLSTM: It is a sequence processing model that consists of two LSTMs. They effectively increase the amount of information available to the network, improving the context available to the algorithm (e.g. knowing what words immediately follow and precede a word in a sentence).

4.1.3 Basics of NLP / Pre-requisites

1. Key Terms
   (a) **NLP**: Natural Language Processing concerns itself with interaction of technology with human languages.
   (b) **Tokenization**: An early step in the NLP process which splits longer strings of text into smaller pieces, or *tokens*.

[1] Everything about Embeddings Embedding converts symbolic representations into meaningful numbers

[2] Morphological tagging is the task of assigning labels to a sequence of tokens that describe them morphologically. As compared to Part-of-speech tagging, morphological tagging also considers morphological features, such as case, gender or the tense of verbs.

(c) **Normalization**: A series of tasks meant to put all text on a level playing field i.e. converting it to lowercase, removing punctuation, expanding contractions, converting numbers to their word equivalents, stripping white space, removing stop words and so on.

  - **Stemming**: Process of eliminating affixes (suffixes, prefixes, infixes, circumfixes) from a word to obtain its stem. For example, *running* becomes *run*.
  - **Lemmatization**: It's related to stemming but is able to capture canonical forms based on the word's lemma (root form). For example, *better* would turn into *good*.

(d) **Corpus**: The latin word for *body* refers to a collection of texts which may be formed of a single language of texts, or multiple. They are generally used for statistical linguistic analysis and hypothesis testing.

(e) **Stop words**: Filter words which contribute little to the overall meaning of text since they are the very common words of the language. For example: *the*, *a* etc.

(f) **Parts-of-speech (POS) Tagging**: It consists of assigning a category tag to the tokenized parts of a sentence such as nouns, verbs, adjectives etc. The category of words is distinguished since they share similar grammatical properties.

(g) **Statistical Language Modeling**: It's the process of building a model which takes *words* as input and assign probabilities to the various sequences that can be formed using them.

(h) **Bag of words**: It's a representation model used to simplify the contents of a selection of text by just reducing the words to their frequency.

(i) **n-gram**: It focuses on preserving contagious sequences of N items from the text selection.

2. A framework for NLP

(a) **Data Collection or Assembly**: Building the corpus

(b) **Data Preprocessing**: Perform operations on the collected corpus which consists of tokenization, normalization, substitution (noise removal).

(c) **Data Exploration & Visualization**: Includes visualizing word counts and distributions, generating wordclouds, performing distance measures.

(d) **Model Building**: Choosing the language models (FSM, MM), classifiers and sequence models (RNNs, LSTMs).

(e) **Model Evaluation**

3. Data Representation

   (a) We need to encode text in a way that can be controlled by us using a statistical classifier.

   (b) We go from a set of categorical features in text: words, letters, POS tags, word arrangement, order etc to a series of *vectors*.

   (c) **One-hot Encoding** (Sparse Vectors) :

   - Each word, or token corresponds to a vector element.
   - Result of one-hot encoding is a sparse matrix, that is, for a corpus containing a lot of tokens, representing a small subset of them would lead to a lot of zero vectors which would consume a large amount of memory.
   - One more drawback is that while it contains the information regarding the presence of a certain word, it lacks positional information so making sense of the tokens is not an option. For example, *Kate hates Alex* is the same as *Alex hates Kate*.
   - Variants of one-hot encoding are *bag-of-words*, *n-gram* and *TF-IDF* representations.

   (d) **Dense Embedding Vectors**:

   - The information of the semantic relationship between tokens can be conveyed using manual or learned POS tagging that determines which tokens in a text perform what type of function. (noun, verb, adverb, etc)
   - This is useful for *named entity recognition*, i.e. our search is restricted to just the nouns.
   - But if one represents *features*[3] as dense vectors i.e. with core features embedded into an embedding space of size d dimensions, we can compress the number of dimensions used to represent a large corpus into a manageable amount.
   - Here, each feature no longer has its own dimension but is rather mapped to a vector.

[3] They are the different categorical characteristic of the given data. For example, it could be *grammatical* classes or some *physical* features. It is context and result dependent. Then for each token, a weight is assigned to it with respect to each feature.

4. Word Representation

5. Subword models

   (a) **Purely Character-level models**: In character-level modes, word embeddings[4] can be composed of character embeddings which have several advantages. *Character-level* models are needed because:

      - Languages like Chinese don't have *word segmentations*.
      - For languages that do have, they segment in different ways.
      - To handle large, open, informal vocabulary.
      - Character level model can generate embeddings for *unknown* words.
      - Similar spellings share similar embeddings

   (b) **Subword-models**: TBD???

4.1.4 Introduction

Morphosyntactic tagging accuracy has improved due to using BiLSTMs to create *sentence-level context sensitive encodings*[5] of words which is done by creating an initial context insensitive word representation[6] having three parts:

1. A dynamically trained word embedding

2. A fixed pre-trained word-embedding, induced from a large corpus

3. A sub-word character model, which is the final state of a RNN model that ingests one character at a time.

In such a model, sub-word character-based representations only interact via subsequent recurrent layers. To elaborate, context insensitive representations would normalize words that shouldn't be, but due to the subsequent BiLSTM layer, this would be overridden. This behaviour differs from traditional linear models.[7]

This paper aims to investigate to what extent having initial subword and word context insensitive representations affect performance. It proposes a

[4] A word embedding is a learned representation for text where words that have the same meaning have a similar representation.

[5] Graph based Neural Dependency Parser

[6] POS Tagging with BiLSTM

[7] *Fast POS Tagging: SVM Approach

hybrid model based on three models- context sensitive initial character and word models and a meta-BiLSTM model which are all trained synchronously.

On testing this system on 2017 CoNLL data sets, largest gains were found for morphologically rich languages, such as in the Slavic family group. It was also benchmarked on English PTB(?) data, where it performed extremely well compared to the previous best system.

4.1.5 Related Work

1. An excellent example of an accurate linear model that uses both word and sub-word features.[7] It uses context sensitive n-gram affix features.

2. First Modern NN for tagging which initially used only word embeddings[8], was later extended to include suffix embeddings.[9]

3. TBD TBD

4. This is the jumping point for current architectures for tagging models with RNNs.[6]

5. Then [5] showed that subword/word combination representation leads to state-of-the-art morphosyntactic tagging accuracy.

4.1.6 Models

1. Sentence-based Character Model In this model, a BiLSTM is applied to all characters of a sentence to induce fully context sensitive initial word encodings. It uses sentences split into UTF8 characters as input, the spaces between the tokens are included and each character is mapped to a dynamically learned embedding. A forward LSTM reads the characters from left to right and a backward LSTM reads sentences from right to left.

   For an n-character sentence, for each character embedding $(e_1^{char}, ..., e_n^{char})$, a BiLSTM is applied:

   $$f_{c,i}^0, b_{c,i}^0 = BiLSTM(r_0, (e_1^{char}, ..., e_n^{char}))_i$$

   For multiple layers(l) that feed into each other through the concatenation of previous layer encodings, the last layer has both forward

[8] Unified architecture for NLP

[9] NLP(almost) from Scratch

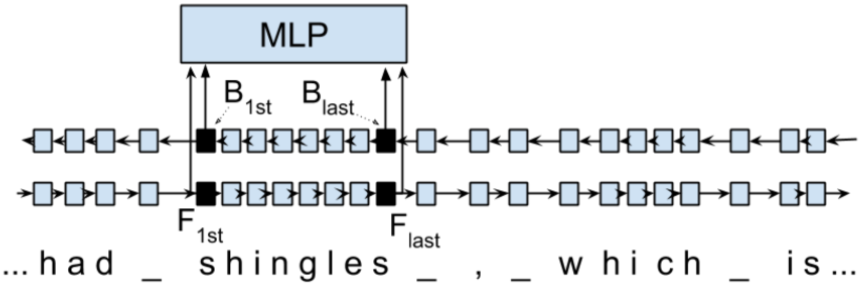


Figure 1: Sentence-based Character Model: The representation for the token *shingles* is the concatenation of the four shaded boxes.

$(f^l_{c,l}, ..., f^l_{c,n})$ and backward $(b^l_{c,l}, ..., b^l_{c,n})$ output vectors for each character.

To create word encodings, relevant subsets of these context sensitive character encodings are combined which can then be used in a model that assigns morphosyntactic tags to each word directly or via subsequent layers. To accomplish this, the model concatenates upto four character output vectors: the {*forward, backward*} output of the {*first, last*} character in the token $T = (F_{1st}(w), F_{last}(w), B_{1st}(w), B_{last}(w))$ which are represented by the four shaded box in *Fig. 1.*

Thus, the proposed model concatenates all four of these and passes it as input to an multilayer perceptron (MLP):

$$g_i = concat(T)$$

$$m_i^{chars} = MLP(g_i)$$

A tag can then be predicted with a *linear classifier* that takes as input m_i^{chars}, applies a *softmax* function and chooses for each word the tag with highest probability.

2. Word-based Character Model To investigate whether a sentence sensitive character model (*Fig.1*) is better than a model where the context is restricted to the characters of a word, (*Fig.2*) which uses the final state of a unidirectional LSTM, combined with the attention mechanism of (ADD REF: cao rei) over all characters.

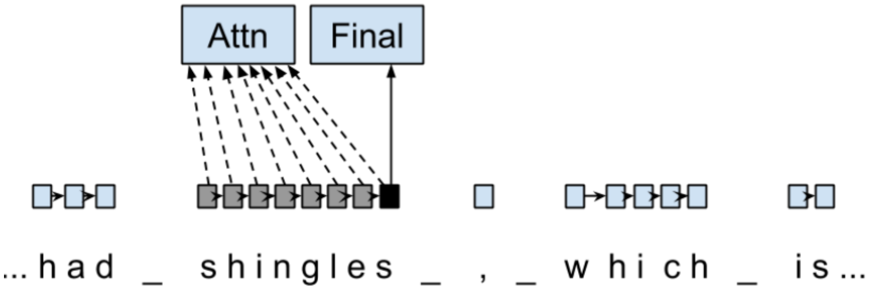


Figure 2: Word-based Character Model: The token is represented by concatenation of attention over the lightly shaded boxes with the final cell (dark box).

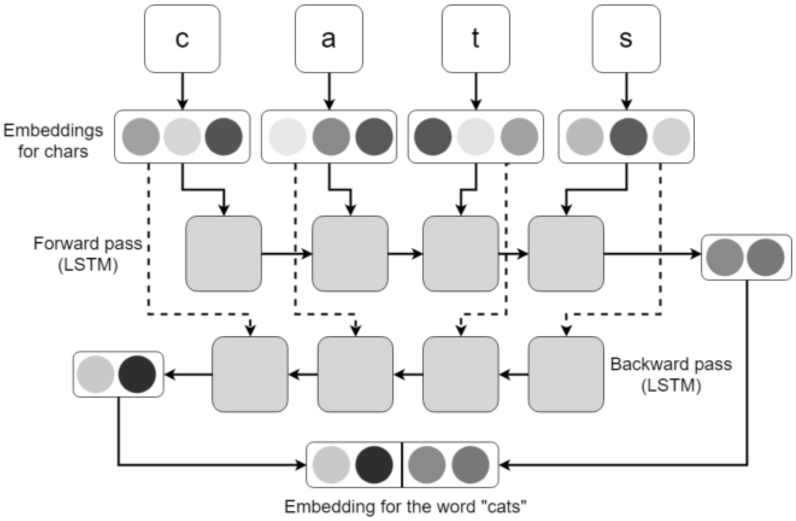


Figure 3: BiLSTM variant of Character-level word representation

3. Sentence-based Word Model The inputs are the words of the sentence and for each of the words, we use pre-trained word embeddings $(p_1^{word}, ..., p_n^{word})$ summed with a dynamically learned word embedding for each word in the corpus $(e_1^{word}, ..., e_n^{word})$:

   $$in_i^{word} = e_i^{word} + p_i^{word}$$

   The summed embeddings in_i are passed as input to one or more BiLSTM layers whose output $f_{w,i}^l, b_{w,i}^l$ is concatenated and used as the final encoding, which is then passed to an MLP:

   $$o_i^{word} = concat(f_{w,i}^l, b_{w,i}^l)$$

   $$m_i^{word} = MLP(o_i^{word})$$

   The output of this BiLSTM is essentially the Word-based Character Model before tag prediction, with the exception that the word-based character encodings are excluded.

4. Meta-BiLSTM: Model Combination If each of the character or word-based encodings are trained with their own loss and are combined using an additional meta-BiLSTM model, optimal performance is obtained. The meta-biLSTM model concatenates the output of context sensitive character and word-based encoding for each word and puts this through another BiLSTM to create an *additional* combined context sensitive encoding. This is followed by a final MLP whose output is passed to a linear layer for tag prediction.

   $$cw_i = concat(m_i^{char}, m_i^{word})$$

   $$f_{m,i}^l, b_{m,i}^l = BiLSTM(r_0, (cw_0, ..., cw_n))_i$$
   $$m_i^{comb} = MLP(concat(f_{m,i}^l, b_{m,i}^l))$$

5. Training Schema Loss of each model is minimized independently by separate optimizers with their own hyperparameters which makes this a multi-task learning model and hence a schedule must be defined in which individual models are updated. In the proposed algorithm, during each epoch, each of the models are updated in sequence using the entire training data.

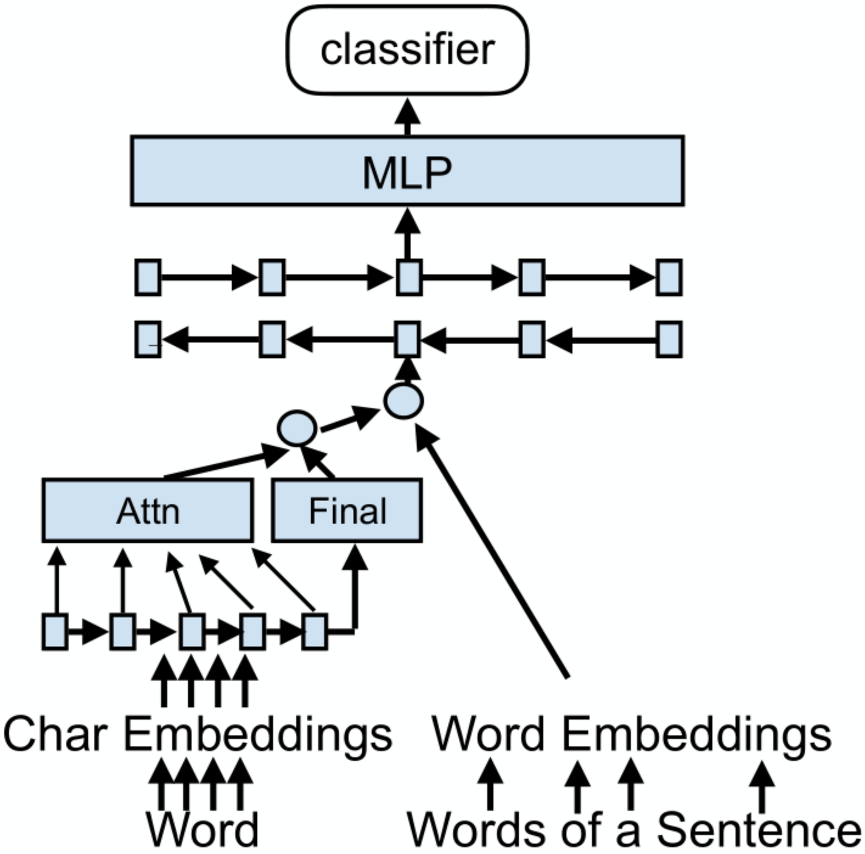


Figure 4: Tagging Architecture of Word-based Character Model and Sentence-based Word Model

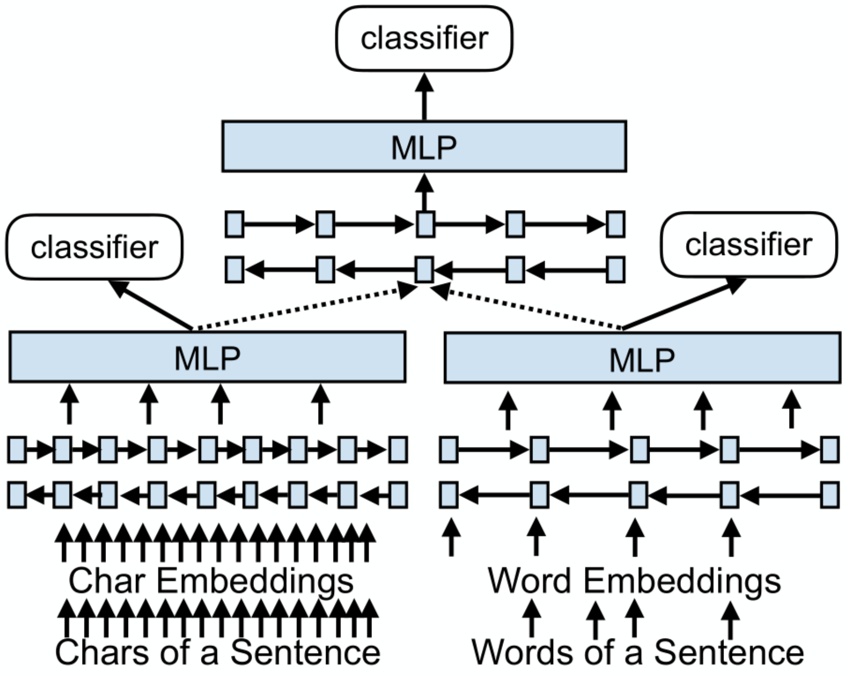


Figure 5: Tagging Architecture of Meta-BiLSTM. Data flows along the arrows and the optimizers minimize the loss of the classifiers independently and backpropogate along the bold arrows.

```
Data: train-corpus, dev-corpus
/* The following models are defined in §3. */
Input: char-model, word-model, meta-model
/* Model optimizers */
Input: char-opt, word-opt, meta-opt
/* Results are parameter sets for each model. */
Result: best-char, best-word, best-meta
/* Initialize parameter sets (cf. Table 1) */
Initialize($pa_c, pa_w, pa_m$)
/* Iteration on over training corpus. */
for epoch = 1 to MAX do
    /* Update character model. */
    char-logits, char-preds = char-model(train-corpus, $pa_c$)
    $pa_c$ = char-opt.update(char-preds, train-data)
    /* Update word model. */
    word-logits, word-preds = word-model(train-corpus, $pa_w$)
    $pa_w$ = word-opt.update(char-preds, train-data)
    /* Update Meta-BiLSTM */
    meta-preds = meta-model(train-corpus, $pa_c, pa_w, pa_m$)
    $pa_m$ = meta-opt.update(train-corpus, meta-preds)
    /* Evaluate model due to dev set accuracy. */
    F1 = DevEval($par_c, par_w, par_m$)
    /* Keep the best model. */
    if F1 > best-F1 then
        best-char = $pa_c$; best-word = $pa_w$ best-meta = $pa_m$; best-F1 = F1
    end
end
```

In terms of model selection, after each epoch, the algorithm evaluates the tagging accuracy of the development set and keeps the parameters of the best model. Accuracy is measured using the meta-BiLSTM tagging layer, which requires a forward pass through all three models. Only the meta-BiLSTM layer is used for model selection and test-time prediction.

The training is synchronous as the meta-BiLSTM model is trained in tandem with the two encoding models, and not after they have converged. When the meta-BiLSTM was allowed to back-propagate through the whole network, performance degraded regardless of the number of loss functions used. Each language could in theory used separate hyperparameters but identical settings for each language works well for large corpora.

4.1.7 Experiments and Results

1. Experimental Setup The word embeddings are initialized with zero values and the pre-trained embeddings are not updated during training. The dropout[10] used on the embeddings is achieved by a single dropout mask and dropout is used on the input and the states of the LSTM.

Model	Parameter	Value
C,W	BiLSTM Layers	3
M	BiLSTM Layers	1
CWM	BiLSTM size	400
CWM	Dropout LSTM	0.33
CWM	Dropout MLP	0.33
W	Dropout Embeddings	0.33
C	Dropout Embedding	0.5
CWM	Nonlinear Activation Fn (MLP)	ELU

   TODO Add two remaining tables

2. Data Sets

3. POS Tagging Results

4. POS Tagging on WSJ

5. Morphological Tagging Results

[10] Dropping out units (hidden and visible) in a neural network, helps prevent the network from overfitting.

4.1.8 Ablation Study (Takeaways)

- **Impact of the training schema**: Separate optimization better than Joint optimization
- **Impact of the Sentence-based Character Model**: Higher accuracy than word-based character context
- **Impact of the Meta-BiLSTM Model Combination**: Combined model has significantly higher accuracy than individual models
- **Concatenation Strategies for the Context-Sensitive Character Encodings**: Model bases a token encoding on both forward and backward character representations of both first and last character in token. (*Fig. 1*) ....
- **Sensitivity to Hyperparameter Search**: With larger network sizes, capacity of the network increases, but it becomes prone to overfitting. Future variants of this model might benefit from higer regularization.
- **Discussion**: TODO Proposed modifications

4.1.9 Conclusions

4.1.10 Readings and Resources

1. Pytorch: Beginner Guide, Detailed Guides, Notebook form
2. Math: Matrix Calculus, Book
3. Basics:
    - Python
    - Jupyter
    - Numpy, Numpy 2
    - Pandas, Pandas 2
    - Matplotlib, Matplotlib 2
    - Seaborn
    - Overview
4. Interactive Tutorials on Weight Initialization, Different Optimizers
5. Rougier's Bits

  - Matplotlib Tutorial, Matplotlib Cheatsheets
  - Numpy Tutorial, From Python to Numpy, 100 Numpy Exercises
  - Python & OpenGL for Scientific Visualization, Scientific Visualization

6. NLP: Best Practices, DL Techniques for NLP

7. BiLSTM: Improving POS tagging

8. Implementation of the paper

4.2 Creating a blog using ox-hugo, org mode and github pages

I was going to make a post explaining how I made this blog but it was rendered pretty useless by this. So yeah, I might archive this later.

1. Install hugo from your package manager.

2. Create a new site:

   ```
   hugo new site blog
   ```

3. Add a theme:

   ```
   cd blog
   git init
   git submodule add <theme_url> themes/<name>
   ```

4. Install ox-hugo in emacs

   ```
   ;; goes in packages.el
   (package! ox-hugo)

   ;; goes in config.el
   (use-package ox-hugo
     :after ox)
   ```

5. TODO Explain the process of content and properties, tags etc.

6. Export

7. Config.toml (theme, title, url, publishdir, etc)
8. Run server, check localhost.
9. Push
10. Go to GitHub repository Settings > GitHub pages. Select /docs in Source.
11. Voila!

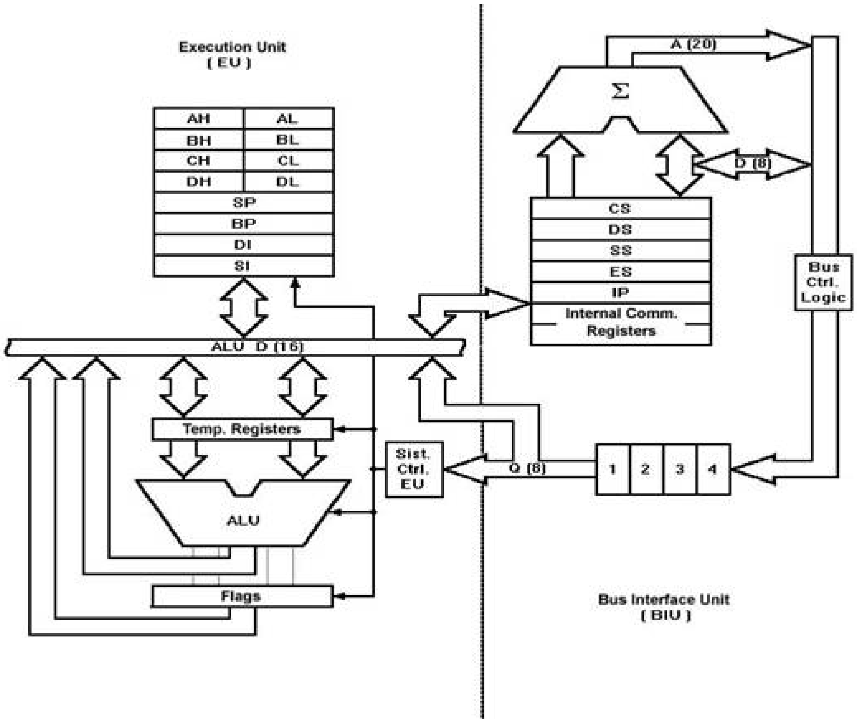
5 Readings

6 Resources

7 Notes

7.1 Microprocessors and Interfacing

7.1.1 Programmer's Model 8086



7.2 Differential Geometry

7.2.1 Theory of Space Curves

1. Representation of space curves

   - Level Curve: f(x,y,z) = C
   - From level curves to parametrized curves: $y = x^2 <------> \gamma(t) = (\gamma_1(t), \gamma_2(t))$ Taking $\gamma_1(t) = t$, we get $\gamma_2(t) = t^2$ hence the parametrization is $\gamma(t) = (t, t^2)$
   - **NOTE:** Check if domain of x satisfies domain of t or not. That is, the same parametrisation can be represented as $(t^2.t^4)$ or (t^3, t^6) but only the latter is a correct representation.
   - From parametrized curves to level curves: $\gamma(t) = (cos^3 t, sin^3 t)$ <——> F(x,y)=C; Using $sin^2 t + cos^2 t = 1$ we get, $x^{2/3} + y^{2/3} = 1$ as the level curve.

2. Unique Parametric representation

   - Class 'm' → f is m-differentiable
   - A curve is *smooth* if $\frac{d^n f}{dt^n}$ exists for all n ≥ 1 and t ∈ (α,β)
   - A function f is *analytic* if it is single valued and of class ∞
   - A function is *regular* if it is differentiable and derivative is non-zero (f dot ≠ 0)
   - A *regular f* of class *m* can also be called a ***path*** of class *m.*
   - **NOTE:** A point of a parametrized curve can have multiple tangents.

3. Arc-length

   - Arc-length of a curve γ is given by the function $s(t) = \int_{t_0}^{t} ||\dot{\gamma}(u)|| du$
   - Speed: $||\dot{\gamma}(t)||_t$ and a curve is unit-speed curve if its magnitude is 1 for all *t.*
   - For γ being a unit speed curve, $\ddot{\gamma}$ is zero or perpendicular to $\dot{\gamma}$ i.e. $\ddot{\gamma}.\dot{\gamma} = 0$
   - If γ is a regular curve, then its arclength S at any point of γ is a smooth function of t.
   - Reparametrization: $\overline{\gamma} : (\overline{\alpha}, \overline{\beta}) \to R^n <=> \gamma : (\alpha, \beta) \to R^n$ exists iff ∃ a smooth function ϕ: $(\overline{\alpha}, \overline{\beta}) \to (\alpha, \beta)$ such that its inverse ϕ^{-1} is also smooth.

- A *unit speed reparametrization* exists for a curve iff it is *regular.*

4. Tangent and Osculating Plane

   - Assuming γ is a class ≥ 1 i.e. it has a power series expansion,

   $$\gamma(u) = \gamma(u_0+h) = \gamma(u_0)+\frac{h}{1!}\dot{\gamma}(u_0)+\frac{h^2}{2!}\ddot{\gamma}(u_0)+...+\frac{h^n}{n!}\gamma^n(u_0)+O(h^n)$$

   where $h = u - u_0$

   - Let γ be class m $\geq$ 2 and (P,Q) be points limiting position of a plane that contains tangential line at P and passes through Q as Q $\rightarrow$ P is defined as the *osculating plane.*
   - **Tangent line:** $\vec{R}(t) = \vec{r}(u_0) + t\vec{r'}(u_0)$ at u_0
   - **Osculating Plane:** $[\vec{R}-r\vec{(0)}, r'\vec{(0)}, r''\vec{(0)}] = 0$ where $\vec{R} = (X, Y, Z)$ gives the equation of the OP (here $\vec{r''}(0) \neq 0$). The product inside the box is *scalar triple product.* Also, the OP passes through the unit vector of the curve and is perpendicular to the unit binormal vector.
   - Note that for smallest k $\geq$ 2 such that $r^{\vec{(k)}} = 0$, the last term in the box is replaced by $\vec{r'}^{(k)}(0)$

5. Principal normal and binormal

   - **Normal Plane:** $\vec{t}(0).(\vec{R} - \vec{r}(0)) = 0$ It is perpendicular to the tangent line and is spanned by *n,b*
   - **Principal Normal Vector:** For m $\geq$ 1, $\vec{n} = \frac{\vec{r''}(0)}{||\vec{r''}(0)||}$
   - **Unit Binormal Vector:** $\vec{b} = \vec{t} \times \vec{n}$
   - OP: b.(R-r)
   - NP: t.(R-r)
   - RP: n.(R-r)

6. Curvature and Torsion

   - For a *unit speed curve* or *arc length parametrized* curve γ(t), the curvature κ(t) is defined as $||\ddot{\gamma}(t)||$ (1)
   - For a *regular* curve γ(t) **in** R^3, $\kappa = \frac{||\ddot{\gamma}\times\dot{\gamma}||}{||\dot{\gamma}^3||}$

- For a unit speed curve γ, *unit tangent vector* $\hat{t} = \dot{\gamma}$ and for $\kappa \neq 0$, *unit normal vector* is given by $\hat{n}(s) = \frac{\dot{\hat{\gamma}}(s)}{\kappa(s)}$ since (1). And *unit binormal vector* can be given by $\hat{b} = \hat{t} \times \hat{n}$
- **Orthonormal Basis** of a curve is given by $\{\hat{t}, \hat{n}, \hat{b}\}$
- Now b is given by t × n , hence $\dot{b} = \dot{t} \times n + t \times \dot{n}$, since $\dot{b}$ has to be perpendicular to t and b, $\implies$ $\ddot{b}||n$, therefore $\boxed{\dot{b} = -\tau n}$ **iff** $\kappa \neq 0$.
- Torsion measures the arc rate of turning of osculating plane.
- For a regular curve γ in R^3 with $\kappa \neq 0$, the *torsion* is given by

$$\tau = \frac{(\dot{\gamma} \times \ddot{\gamma}).\dddot{\gamma}}{||\dot{\gamma} \times \ddot{\gamma}||^2}$$

- Also, *radius of curvature* ρ is inverse of curvature.
- Finally, tying it all together is the *Serret-Frenet formula* (arc length parameter): $\begin{bmatrix} \dot{t} \\ \dot{n} \\ \dot{b} \end{bmatrix} = \begin{bmatrix} 0 & \kappa & 0 \\ -\kappa & 0 & \tau \\ 0 & -\tau & 0 \end{bmatrix} \begin{bmatrix} t \\ n \\ b \end{bmatrix}$

7. Behaviour of a curve near one of its points

- For a regular curve of class m $\geq$ 2 with nonvanishing curvature, the curve is *planar* iff τ=0 everywhere.
- For an analytic curve with arc length parameter, as s $\to$ 0, a new parametrization for small s can be defined as:

$$X = s - \frac{\kappa^2 s^3}{6} - \frac{\kappa\kappa' s^4}{8} + o(s^4)$$

$$Y = \frac{\kappa s^2}{2} + \frac{\kappa' s^3}{6} + \frac{\kappa'' - \kappa\tau - \kappa^3}{24} s^4 + o(s^4)$$

$$Z = \frac{\kappa\tau}{6} s^3 + \frac{2\kappa'\tau + \kappa\tau'}{24} s^4 + o(s^4)$$

- Here the o notation represents that for f = o(g), as s $\to$ 0, $lim\frac{f(s)}{g(s)} = 0$
- From previous theorem:
  (a) $\kappa(0) = \lim_{s\to 0} \frac{2Y}{X^2}$
  (b) $\tau(0) = \lim_{s\to 0} \frac{3Z}{XY}$

(c) For $P = \vec{r}(0), Q = \vec{r}(s)$, the length of chord

$$PQ = s(1 - \frac{\kappa^2 s^2}{24}) + o(s^3)\mathrm{A}s(1 - \frac{\kappa^2 s^2}{24})o(s^3)$$

If f(t)=g(t)+o(t), then as t → 0, it can be written as f(t)˜g(t)o(t)

- The length of common perpendicular between tangents at two nearby points of $\vec{r}(s)$ at arcual distance s is approximately $d = \frac{\kappa\tau s^3}{12}$. This is the shortest distance between tangents at nearby points of r(s).

8. Contact between curves and surface

- For a surface S: F(x,y,z)=0 and a parametrized curve C: $\vec{r}(u) =$ (f(u),g(u),h(u)), let P be a point on C. P lies on S iff F(f(P),g(P),h(P))=0.
- Let ϕ(u) = F(f(u),g(u),h(u)) for any parameter value u. Then P lies on S iff ϕ(u$_0$)=0.
- Assuming F and $\vec{r}$ are of class m for sufficiently large m, then ϕ(u) has a taylor expansion where $\frac{O(h^{n+1})}{h^{n+1}}$ is bounded as h → 0.
- Definition: Surface S and a parametrized curve C has an *n-point contact* (or contact of order n) at P if $\phi(u_0) = \phi'(u_0) = ... = \phi^{(n-1)}(u_0) = 0$ and $\phi^{(n)}(u_0) \neq 0$
- If S and C have a contact of order 1 at P then it is called a *simple intersection* of S and C.
- If P is in n-point contact of S and C, then S and C intersect at P in n coincidental points.
- Condition for *n-point contact* at P is invariant under a change of parameter.
- Osculating Plane at P of $\vec{r}$ has atleast a 3-point contact with $\vec{r}$ at P.

9. Osculating circle (circle of curvature)

- For a regular curve $\vec{r}(s)$ of class m ≥ 2, let $P = \vec{r}(0)$ and $P_i = \vec{r}(s_i), i = 1, 2, 3$ be 3 non collinear points near P on the curve. Then there is a unique circle through all P_i. The limiting circle, if existent, for all $P_i \to P$ is called *osculating circle* of r(s) at P.
- Center of OC (c) is called *centre of curvature* of r(s) at P while its radius $\rho(0)$ is called radius of curvature. Also, the OC lies in the OP.

- Theorem: $\rho(0) = \frac{1}{\kappa(0)}$, $\vec{c}(o) = \vec{r}(0) + \rho(0)\vec{n}(0)$
- OC does not exist at points where curvature vanishes and OC of a circle is the same circle itself.

10. Osculating Sphere

    - Definition: For a regular path r(s) of class m $\geq$ 2, assuming P = r(0) and $\kappa(0)\tau(0) \neq 0$, a sphere which has atleast a 4-point contact with r(s) at P is called *osculating sphere* at P on r.
    - ρ(s)= $\frac{1}{\kappa(s)}$ is called radius of curvature and σ(s)= $\frac{1}{\tau(s)}$ is called radius of torsion of r(s)
    - Theorem: OS at P on r is given by $|\vec{c} - \vec{R}|^2 = R^2$ where $R = \sqrt{\rho(0)^2 + \sigma(0)^2\rho'(0)^2}$ and $\vec{c} = \vec{r}(0) + \rho(0)\vec{n}(0) + \sigma(0)\rho'(0)\vec{b}(0)$ where c and R are COSC and ROSC to r(s) at r(0)
    - Centre of OS lies in the normal plane of r(s) as $c - r(0)$ is a linear combination of n(0) and b(0)
    - If κ is constant then ROC=ROSC and COC=COSC. In particular, if r is a circle, then its its own OC and is a great circle of the OS.

11. Locus of centres of spherical curvature

    - Since COSC at r(s) is $c(s) = r(s) + \rho(s)n(s) + \sigma(s)\rho'(s)b(s)$, it moves along a path as s varies. For this path, SFF, κ, τ can be calculated and will be denoted with subscript c.
    - Assuming τ(s)>0,

      (a) $c'(s) = (\frac{\rho(s)}{\sigma(s)} + \frac{d(\sigma(s)\rho'(s))}{ds})b(s)$

      (b) For a regular c(s), unit tangent vector is $t_c(s) = eb(s)$

      (c) $\frac{ds_c}{ds} = |\frac{\rho(s)}{\sigma(s)} + \frac{d(\sigma(s)\rho'(s))}{ds}|$

      Here e is 1 if ds$_c$/ds > 0, -1 ow. Also $e = t_c(s).b(s)$
    - Also on differentiating,

      (a) $\kappa_c(s) = \frac{\tau(s)}{\frac{ds_c}{ds}}$ or κ(s)= $-\tau_c(s)e\frac{ds_c}{ds}$

      (b) Which gives $\tau(s)\tau_c(s) = \kappa(s)\kappa_c(s)$
    - Theorem: ROC of center of curvatures (i.e. center of OCs) is given by

$$\rho_1 = [(\frac{\rho^2\sigma}{R^3}\frac{d}{ds}(\frac{\sigma\rho'}{\rho}) - \frac{1}{R})^2 + \frac{\rho'^2\sigma^4}{\rho^2R^4}]^{-1/2}$$

12. Tangent surfaces, involutes and evolutes

    - Definition: Tangent surface to a curve r is union of all tangent lines to r at all its points.
    - Tangent line to r at r(s) is R(u,s) = r(s)+ur'(s)
    - For both varying r and u, one gets the tangent surface.
    - Image of the curve u=u(s) in us-plane gives a curve $r_1(s) = r(s) + u(s)r'(s)$
    - Definition: Involute of r is a curve on the tangent surface of r which meets all generating lines orthogonally at corresponding points.
    - If $r_1(s)$ denotes the pos vector on the involute C$_1$ of a curve C corresponding to its points r(s) then r$_1$(s)=r(s)+(c-s)t(s) for a constant c.
    - For an involute c(s) of a regular path r(s) of class m $\geq$ 2.

    $$\kappa_c^2 = \frac{\tau^2 + \kappa^2}{\kappa^2(c-s)^2}, \tau_c = \frac{\kappa\tau' - \kappa'\tau}{\kappa(c-s)(\tau^2+\kappa^2)}$$

    - Definition: If $\overline{C}$ is an involute of C then C is called an evolute of $\overline{C}$.
    - For a regular curve r(s), evolute is given by $r_1(s) = r(s) + \rho(s)n(s) + \rho(s)cot(\psi(s) + c)b(s)$ where c is a constant and ψ(s) = $\int \tau(s)ds$
    - r(s) has infinitely many evolutes, as c is random constant. For a plane curve, τ = 0.
    - Tangents to two different evolutes corresponding to two constans A and B drawn from the same point of the given curve are inclined to each other at a constant angle A-B.

    $$r_1 = r + \rho\mathbf{n} - \rho tan(\psi + a)\mathbf{b}$$

    Further $\psi = \int \tau ds$ so that ψ'=τ...

7.2.2 First Fundamental Form and Local Intrinsic Properties of a Surface

1. Introduction

- The surfaces are defined similar to curves by an equation of the type F(x,y,z) = 0 or parametrically by expressing x,y,z in terms of two parameters u,v varying over a domain.
- After defining the surface locally, its points are classified as ordinary or singular.
- Then using tangent plane at a point and the surface normal at it, a coordinate system (r_1, r_2, N) at every point of the surface is introduced.
- After that, a certain quadratic differential form known as *first fundamental form* on a surface and direction coefficients are introduced.

2. Definition of a Surface **Definition 1:** Locus of a point P(x,y,z) in E_3 satisfying some restrictions on x,y,z which is expressed by a relation of the type F(x,y,z) = 0.

   This equation is called the *implicit* or the *constraint* equation of the surface which allows for a global study of the surface.

   Definition 2: For parameters u, v taking real values and varying over a domain D, a surface is defined *parametrically* as

   $$x = f(u, v), y = g(u, v), z = h(u, v)$$

   where f, g and h are single valued continuous functions possessing continuous derivatives of r-th order. Such surfaces are called surfaces of class r.

   Parametric representation is useful for local study of surfaces i.e. in the neighbourhood of a point which is a small region **but** it is not unique for a surface. Also, the parameters u and v are called *curvilinear coordinates.*

   Definition 3: For two parametric representations u, v and u', v' of the same surface, any transformation of the form $u' = \phi(u, v)$ and $v' = \psi(u, v)$ relating the two representations is called a *parametric transformation.*

   Definition 4: A parametric transformation is *proper* if:

   (a) ϕ and ψ are single valued functions.

   (b) The Jacobian $\frac{\delta(\phi,\psi)}{\delta(u,v)} \neq 0$ in some domain D.

These conditions are necessary and sufficient for existence of inverse in the neighbourhood of any point in D' which is the domain of *u', v'* corresponding to the domain D of the *u, v* plane.

3. Nature of Points on a Surface **Notation:** For **r** being the position vector of a point on the surface, **r** = (x,y,z), we can take r = r(u,v) as the parametric form of the surface and use $r_1 = \frac{\delta r}{\delta u} = (x_1, y_1, z_1)$ and $r_2 = \frac{\delta r}{\delta v} = (x_2, y_2, z_2)$, similarly we can denote second order derivatives using r_{11}, r_{21} etc.

   Definition 1: If $r_1 \times r_2 \neq 0$ at a point on a surface, then the point is called an *ordinary* point. A point which is not an ordinary point is called a *singularity.*

   Remarks:

   - Considering M = $\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{bmatrix}$ For $r_1 \times r_2 \neq 0$ at an ordinary point, i.e. rank of M is two at that point.
   - If the rank of M is either zero or one, the point on the surface is a singular point.
   - If $r_1 \times r_2 \neq 0$ or equivalently rank of M is two, then *x,y,z* uniquely determine the parameters *u,v* in the neighbourhood of an ordinary point.
   - When only one determinant minor of M is zero, one cannot conclude that the point is a singular point.
   - A *proper* parametric transformation transforms an ordinary point into an ordinary point.
   - Due to geometrical nature of the surface, some singularities continue to be singularities, regardless of the parametric representations, these are called *essential singularities.*
   - There are other singularities depending on the choice of parametric representation which are called *artificial singularities.*

   Example: Consider the circular cone represented by $x = u \ sin\alpha \ cosv$, $y = u \ sin\alpha \ sinv$, $z = u \ cos\alpha$ where α is the semivertical angle of cone with O as origin and OP = u, where P is any point on the cone. Computing M, then at $u = 0$, the determinant of every second order minor is zero, hence it is an essential singularity.

   Example: Taking any point 0 as origin in the plane, $x = u \ cosv$, $y = u \ sinv$, $z = 0$, we get $r_1 \times r_2 = u\mathbf{k}$. Hence it is zero only when $u = 0$

i.e. it is an artificial singularity *since* it arises due to the choice of the parametric coordinates and not due to the nature of the surface.

4. Representation of a Surface For our study of surfaces, we consider only ordinary points. And we consider the entire surface as a collection of parts, each part being given a particular parametrisation and the adjacent parts being related by a *proper* parametric transformation.

   Definition 1: A representation R of a surface S of class r in E_3 is a collection of points in E_3 covered by a system of overlapping parts S_j where each part $\{\{S_j\}$ is given by a parametric equation of class r. Each point lying in the common portion of two parts S_i, S_j is such that the change of parameters from one part to is adjacent is given by a *proper* parametric transformation of class r.

   Note: Since one cannot parameterise the whole surface without introducing artificial singularities, one has to resort to a surface composed of many overlapping parts.

   It is possible to have many representations of the same surface by considering different systems of overlapping parts (S_j), each part is given by a parametric equation of class r.

   Definition 2: For R and R' being two representations of class r of the surface S, they are *equivalent* if the composite family of parts $\{S_j, S'_j\}$ satisfies the condition that for each point P lying in the place of overlap, the change of parameter from S_j to S'_j at P is given by a proper parametric transformation of class r.

   Theorem: The notion of r-equivalence of representations of a surface is an equivalence relation.

   This equivalence relation introduces a partition into the family of surfaces of class r splitting them into mutually disjoint equivalence classes, each class containing the surface equivalent to one another in the above equivalence relation.

   Definition 3: A surface S of class r in E_3 is an r-equivalence class of representations.

   Thus a surface consists of different overlapping portions related to one another by proper parametric transformations and all other surfaces related to the given one by the equivalence relation of class r.

5. Curves on Surfaces For a surface $\mathbf{r}$ = r(u,v), let u = $u(t)$ and v = $v(t)$ be a curve of class s lying in the domain D of the uv-plane. Considering

$\mathbf{r}$ = r[u(t), v(t)] which gives the position vector of a point in terms of a single parameter t such that it is a curve lying on a surface with class equal to the smaller of r and s. The equation u = *u(t)* and v = *v(t)* are called *curvilinear equations* of the curve on the surface.

Definition 1: For $\mathbf{r}$, a given surface of class r, let $v = c$, then position vector $\mathbf{r}$ = r(u,c) is a function of a single parameter t and hence $\mathbf{r}$ = r(u,c) represents a curve lying on the surface $\mathbf{r}$ = r(u,v). This curve is called the *parametric curve* v = constant.

By varying the values of c, a system of parametric curves v = constant is generated and similarly another system is generated by keeping u constant and varying v.

Properties that are a consequence of assuming only ordinary points on the surface:

(a) Through every point of the surface, there passes one and only one parametric curve of each system.

(b) No two curves of the same system intersect.

(c) The curves of the system $u = u_o$ and $v = v_o$ intersect once but not more than once if $(u_o, v_o) \in D$.

(d) The parametric curves of the system u = c_1 and v = c_2 cannot touch each other.

Definition 2: Let u = c_1 and v = c_2, when the constants vary, the whole surface is covered with a net of parametric curves, two of which pass through each point.

Definition 3: Two parametric curves through a point P are *othogonal* if $\mathbf{r}_1.\mathbf{r}_2 = 0$ at P.

6. Tangent Plane and Surface Normal Let $\mathbf{r}$ = r[u(t), v(t)] be a general curve lying on the surface passing through [u(t), v(t)], then the tangent to the curve at any point P on the surface is

$$\frac{dr}{dt} = r_1\frac{du}{dt} + r_2\frac{dv}{dt}$$

Definition 1: Tangent to any curve drawn on a surface is called a tangent line to the surface. The tangents to different curves through P on a surface lie in a plane containing two independent vectors r_1 and r_2 at P called the *tangent plane* at P.

Theorem 1: The equation of a tangent plane at P on a surface with position vector **r** $=$ r(u,v) is either $R = r+ar_1+br_2$ or $(R-r).(r_1\times r_2) = 0$ where a and b are parameters.

Definition 2: The normal to the surface P is a line through P and perpendicular to the tangent plane at P.

Theorem 2: The equation of the normal **N** at a point P on the surface r $=$ r(u,v) is $R = r + a(r_1 \times r_2)$.

Theorem 3: A proper parametric transformation either leaves every normal unchanged or reverses the direction of the normal.

7. General Surface of Revolution **Definiton 1:** A surface generated by the rotation of a plane curve about an axis in its plane is called a *surface of revolution.*

   Theorem 1: The position vector of any point on the surface of revolution generated by the curve [g(u),o,f(u)] in the XOZ plane is

   $$\mathbf{r} = [g(u)cosv, g(u)sinv, f(u)]$$

   where v is the angle of roatation about the z-axis.

Emacs 28.0.50 (Org mode 9.5)