Extracting Breast Cancer Genetic Markers in Pathology Reports using Natural Language Processing

Gopinath Chennupati[1], Kumkum Ganguly[1], Benjamin McMahon[1], Sunil Thulasidasan[1], Jessica Boten[2], Valentina Petkov[2], Lynne Penberthy[2], Xiao-Cheng Wu[3], Paul Fearn[2] and Tanmoy Bhattacharya[1]

[1]Los Alamos National Laboratory
[2]National Cancer Institute
[3]Louisiana Tumor Registry

Table of contents

Introduction

Objectives

1. Given a breast cancer (**C50**) pathology report identify: ER, PR, HER2
    - Semantic (context) word embeddings from electronic pathology reports
    - Employ a Multi-Task Deep Learning algorithm
2. Uncertainty Quantification (UQ) of deep models

Pathology Reports

- Patient data:
  - electronic pathology reports (XML format)
- For example The Estrogen Receptor (VECTOR-CLONE 6F11) is negative in 100% of the tumor cells showing 0 staining ...
- Extract tumor genomic marker information from text

Pathology Reports

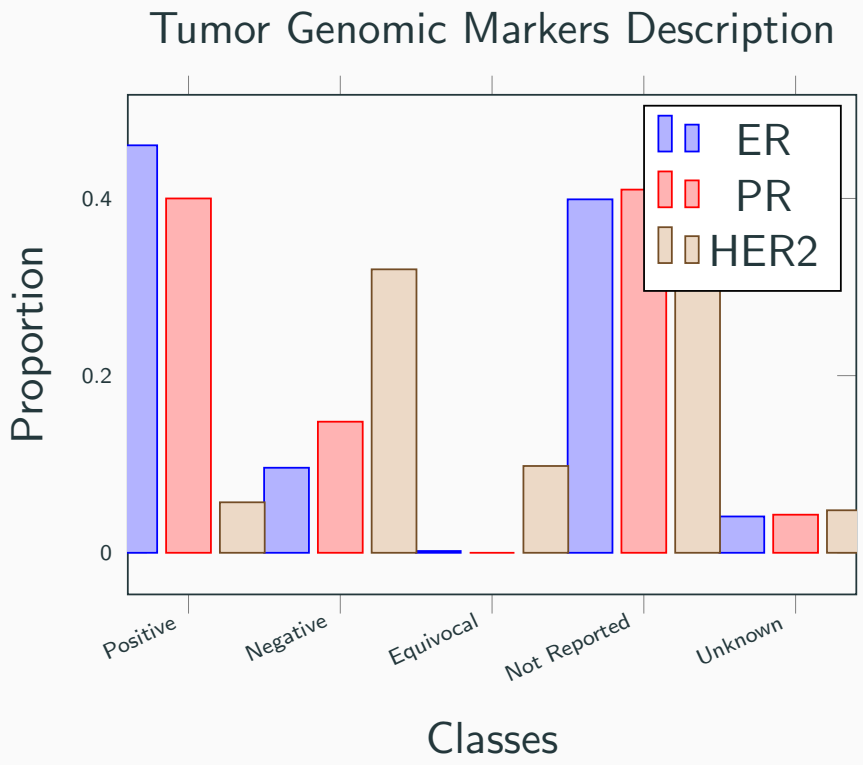
- Patient data:
  - electronic pathology reports (XML format)
- For example The Estrogen Receptor (VECTOR-CLONE 6F11) is negative in 100% of the tumor cells showing 0 staining ...
- Extract tumor genomic marker information from text
- Difficult, because –
  - Text is semi-structured
  - Un-standardized terms, abbreviations and acronyms
    For example, ER (or) estrogen receptor
  - Information may be in different sections of the report, etc.

Tumor Genomic Markers

- Five different classes

Table 1: Count of breast cancer reports per registry

Registry	C50 count
HI	120
KY	118
NM	146
CT	50
Seattle	124
Total	558



- PR has no *Neutral* classes

Classification via Simple NLP Techniques

- Term Frequency Inverse Document Frequency (TF-IDF)

Table 2: ML classifiers with all **962** features extracted using TF-IDF.

Input	Ada Boost	DT	Gaussian Process	Linear SVM	Naive Bayes	Nearest Neigh	MLP	QDA	RBF SVM	RF
ER	50.00	57.14	66.67	61.90	61.90	54.76	**62.05**	42.86	42.86	52.38
HER2	47.62	47.62	54.76	**59.52**	47.62	45.24	**59.52**	54.76	54.76	50.00
PR	38.10	47.62	59.52	57.14	54.76	45.24	**61.90**	42.86	42.86	54.76

Classification via Simple NLP Techniques

- Term Frequency Inverse Document Frequency (TF-IDF)

Table 2: ML classifiers with all **962** features extracted using TF-IDF.

Input	Ada Boost	DT	Gaussian Process	Linear SVM	Naive Bayes	Nearest Neigh	MLP	QDA	RBF SVM	RF
ER	50.00	57.14	66.67	61.90	61.90	54.76	**62.05**	42.86	42.86	52.38
HER2	47.62	47.62	54.76	**59.52**	47.62	45.24	**59.52**	54.76	54.76	50.00
PR	38.10	47.62	59.52	57.14	54.76	45.24	**61.90**	42.86	42.86	54.76

- Multi-layered Perceptrons performed better
- Domain expertise – reduced features (**25**)
- Accuracy 70%.

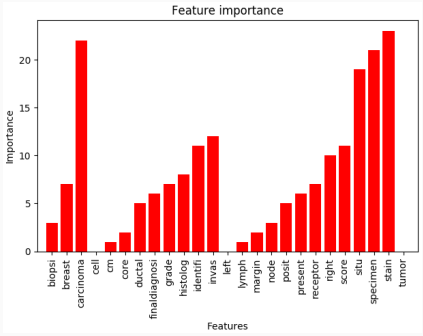


Figure 1: Feature importance

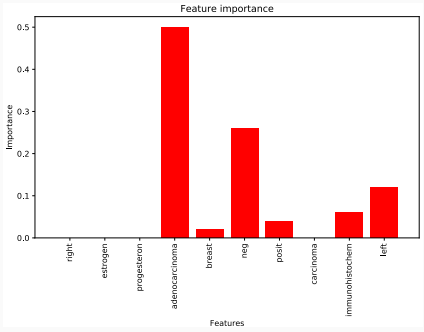


Figure 2: Reduced features

Semantic Word Embeddings

- Feed words to DL algorithm as numeric vectors
- Keep the context (meaning) of a word by estimating the probabilities of other words that are close
- PLOS One Oncology papers to prepare word embeddings

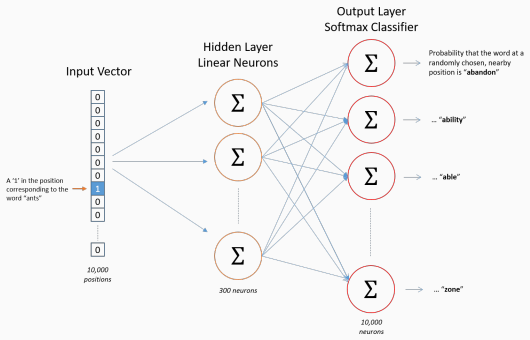


Figure 3: NN for Word2Vec embeddings

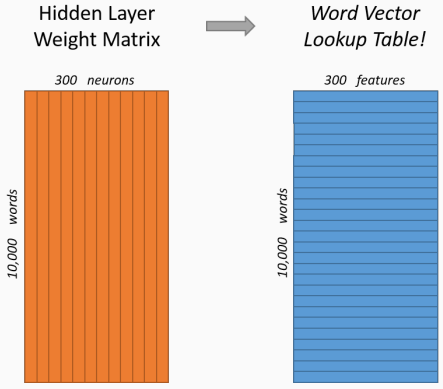


Figure 4: Word vectors – Semantic embeddings

Semantic Word Embeddings

- Feed words to DL algorithm as numeric vectors
- Keep the context (meaning) of a word by estimating the probabilities of other words that are close
- PLOS One Oncology papers to prepare word embeddings

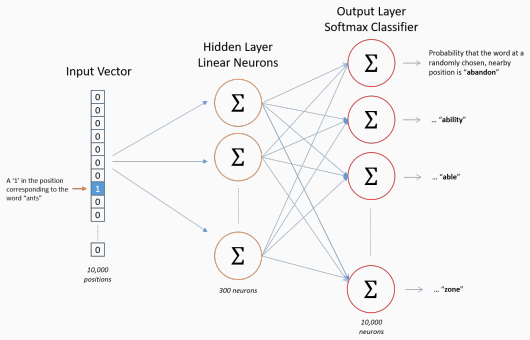


Figure 3: NN for Word2Vec embeddings

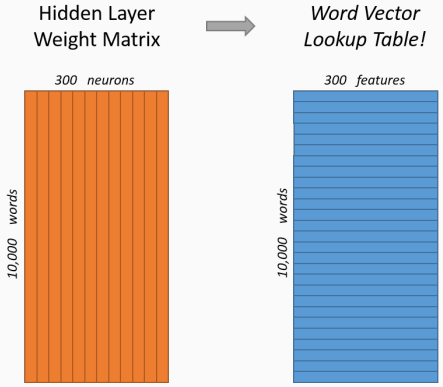


Figure 4: Word vectors – Semantic embeddings

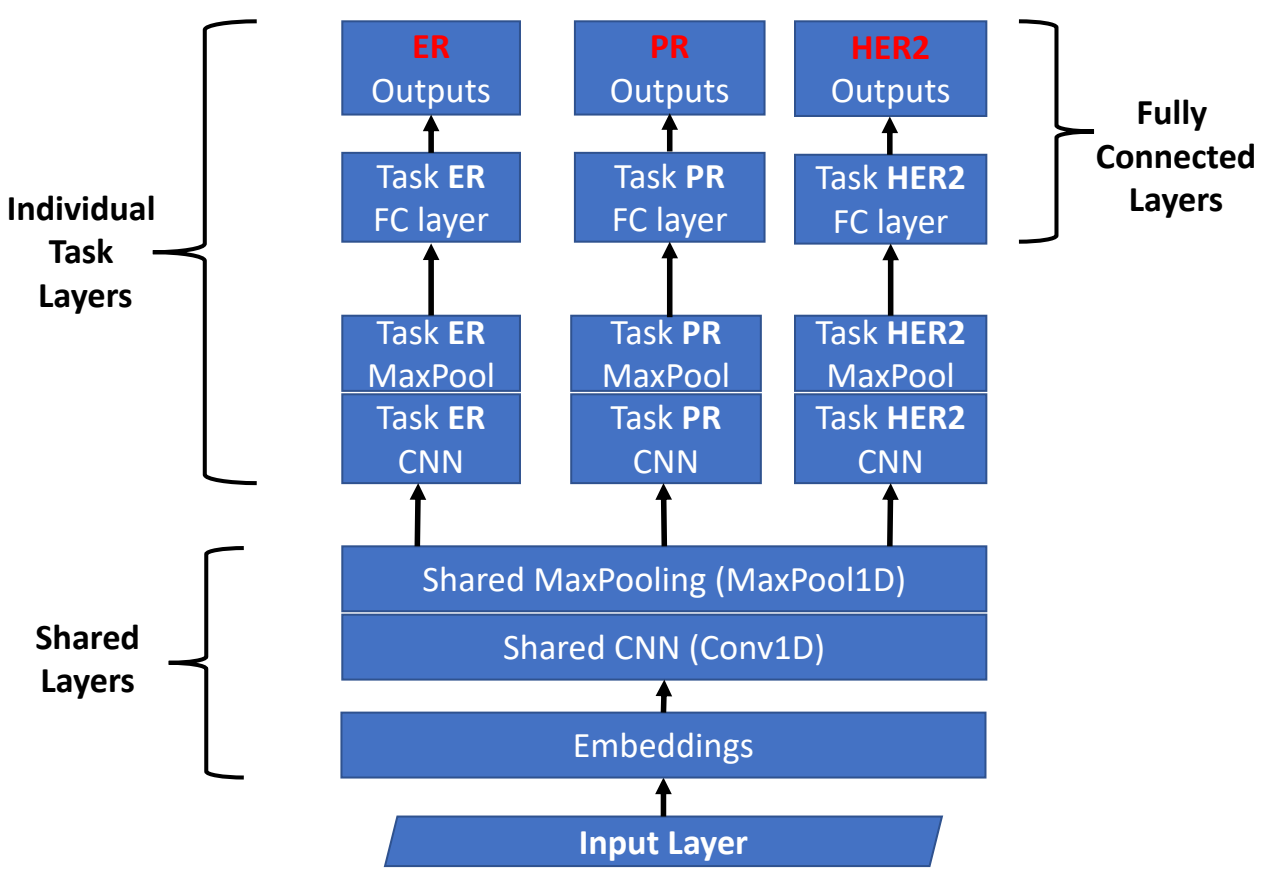
w2v.most_similar('estrogen')
(u'hormone', 0.681),
(u'estrogen', 0.654),
(u'mone', 0.647),
(u'hor', 0.621),
(u'pgr', 0.592)

Figure 5: Top 5 most similar words

Multi-Task Learning (MTL)

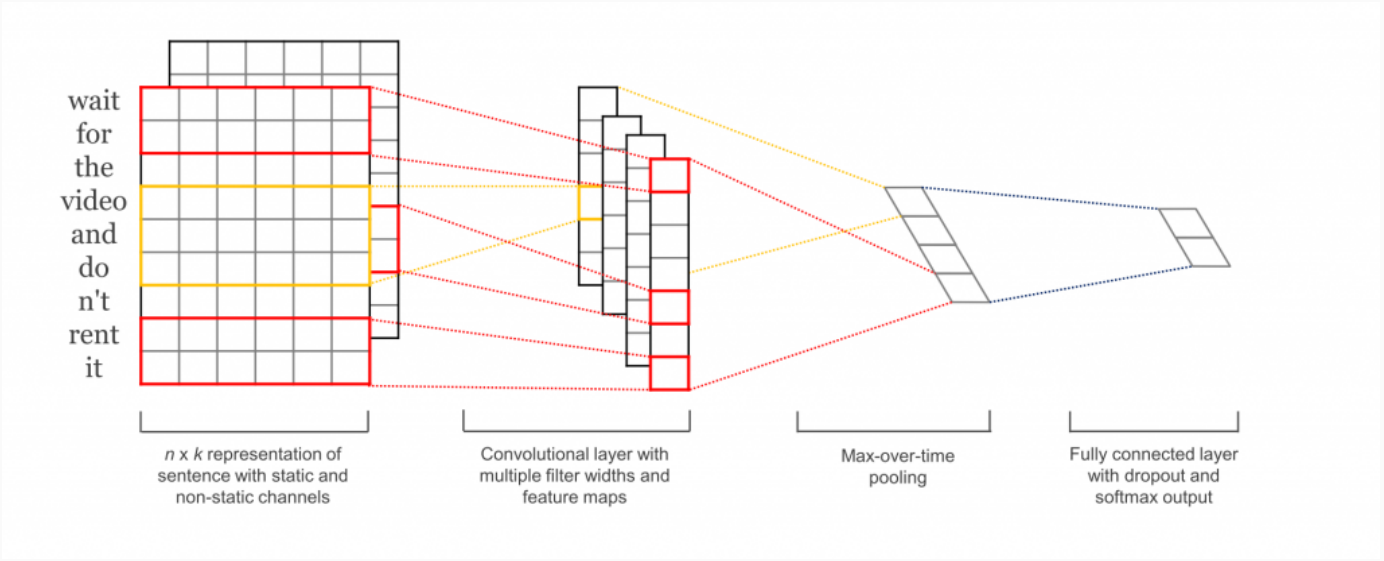
Multi-Task Convolution Neural Network (MT-CNN)

- Three different tasks (ER, PR, HER2)



Convolution Neural Network (CNN)

- An example CNN for Natural Language Processing



- Input text document as a matrix prepared from embeddings
- Convolutions $-->$ Max Pooling $-->$ Fully Connected (FC) layer $-->$ SoftMax

Results

MT-CNN Experiments

- Train, validation and test splits of the data: **0.6**, **0.2** and **0.2**
- Precision (P(C_j)), Recall (R(C_j)) and F-score (F(C_j))

$$P(C_i) = \frac{TP_j}{TP_j + FP_j} \qquad P^{micro} = \frac{\sum_{j=1}^{\mathbf{C}} TP_j}{\sum_{j=1}^{\mathbf{C}} (TP_j + FP_j)} \qquad P^{macro} = \frac{1}{|\mathbf{C}|} \times \sum_{j=1}^{\mathbf{C}} P(C_j)$$

$$R(C_i) = \frac{TP_j}{TP_j + FN_j} \qquad R^{micro} = \frac{\sum_{j=1}^{\mathbf{C}} TP_j}{\sum_{j=1}^{\mathbf{C}} (TP_j + FN_j)} \qquad R^{macro} = \frac{1}{|\mathbf{C}|} \times \sum_{j=1}^{\mathbf{C}} R(C_j)$$

$$F(C_i) = \frac{2 \times P(C_i) \times R(C_i)}{P(C_i) + R(C_i)} \qquad F^{micro} = \frac{2 \times P^{micro} \times R^{micro}}{P^{micro} + R^{micro}} \qquad F^{macro} = \frac{1}{|\mathbf{C}|} \times \sum_{j=1}^{\mathbf{C}} F(C_j)$$

$$micro = \frac{TP_1 + TP_2 + TP_3}{(TP_1 + FP_1) + (TP_2 + FP_2) + (TP_3 + FP_3)} \qquad macro = \frac{1}{3} \times \Big[P(C_1) + P(C_2) + P(C_3) \Big]$$

MT-CNN Experiments

- Train, validation and test splits of the data: **0.6**, **0.2** and **0.2**
- Precision (P(C_j)), Recall (R(C_j)) and F-score (F(C_j))

$$P(C_i) = \frac{TP_j}{TP_j + FP_j} \qquad P^{micro} = \frac{\sum_{j=1}^{\mathbf{C}} TP_j}{\sum_{j=1}^{\mathbf{C}} (TP_j + FP_j)} \qquad P^{macro} = \frac{1}{|\mathbf{C}|} \times \sum_{j=1}^{\mathbf{C}} P(C_j)$$

$$R(C_i) = \frac{TP_j}{TP_j + FN_j} \qquad R^{micro} = \frac{\sum_{j=1}^{\mathbf{C}} TP_j}{\sum_{j=1}^{\mathbf{C}} (TP_j + FN_j)} \qquad R^{macro} = \frac{1}{|\mathbf{C}|} \times \sum_{j=1}^{\mathbf{C}} R(C_j)$$

$$F(C_i) = \frac{2 \times P(C_i) \times R(C_i)}{P(C_i) + R(C_i)} \qquad F^{micro} = \frac{2 \times P^{micro} \times R^{micro}}{P^{micro} + R^{micro}} \qquad F^{macro} = \frac{1}{|\mathbf{C}|} \times \sum_{j=1}^{\mathbf{C}} F(C_j)$$

$$micro = \frac{TP_1 + TP_2 + TP_3}{(TP_1 + FP_1) + (TP_2 + FP_2) + (TP_3 + FP_3)} \qquad macro = \frac{1}{3} \times \Big[P(C_1) + P(C_2) + P(C_3) \Big]$$

- Confidence intervals (CI) are the **(lower, upper)** bounds of performance metrics
- CI are measured with re-sampling the train and test sets at $\alpha = 0.95$

MT-CNN Performance

Table 3: Performance of MTCNN

Metric	Avg. Value (Confidence Interval [CI])
P^{macro}	0.679 (0.507, 0.851)
R^{macro}	0.649 (0.428, 0.870)
F^{micro}	0.719 (0.589, 0.859)
F^{macro}	0.646 (0.436, 0.857)

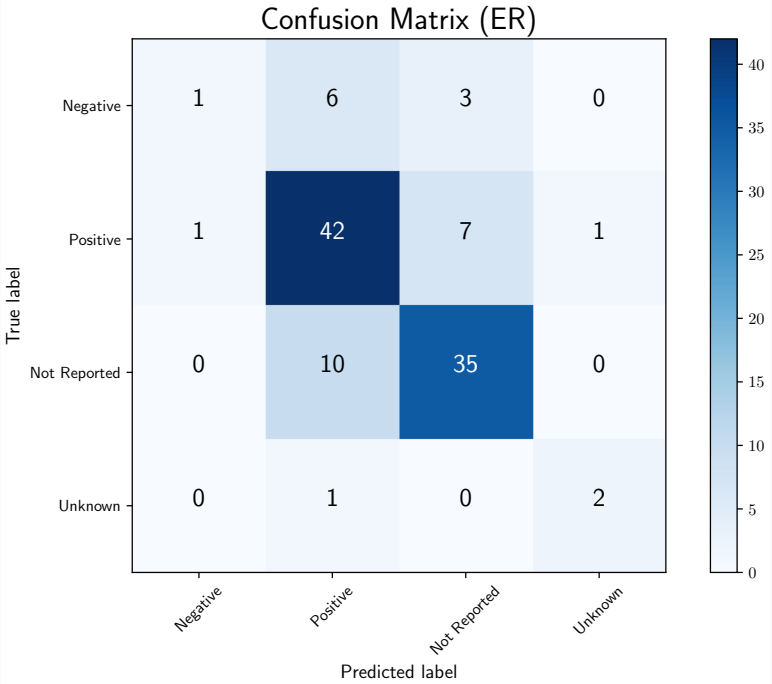


Figure 6: Confusion matrix of predictions

MT-CNN Performance

Table 3: Performance of MTCNN

Metric	Avg. Value (Confidence Interval [CI])
P^{macro}	0.679 (0.507, 0.851)
R^{macro}	0.649 (0.428, 0.870)
F^{micro}	0.719 (0.589, 0.859)
F^{macro}	0.646 (0.436, 0.857)

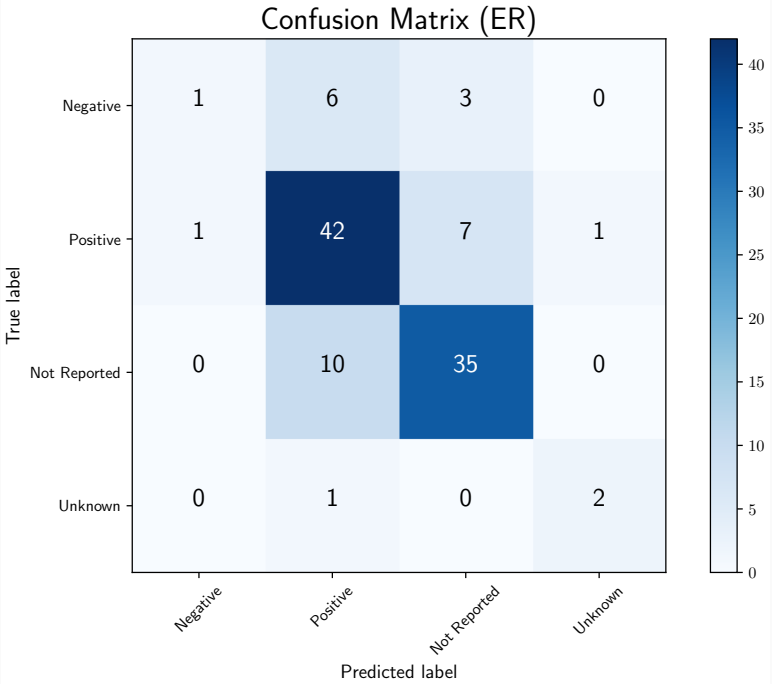


Figure 6: Confusion matrix of predictions

- What contributes to the prediction?

Model Visualizations

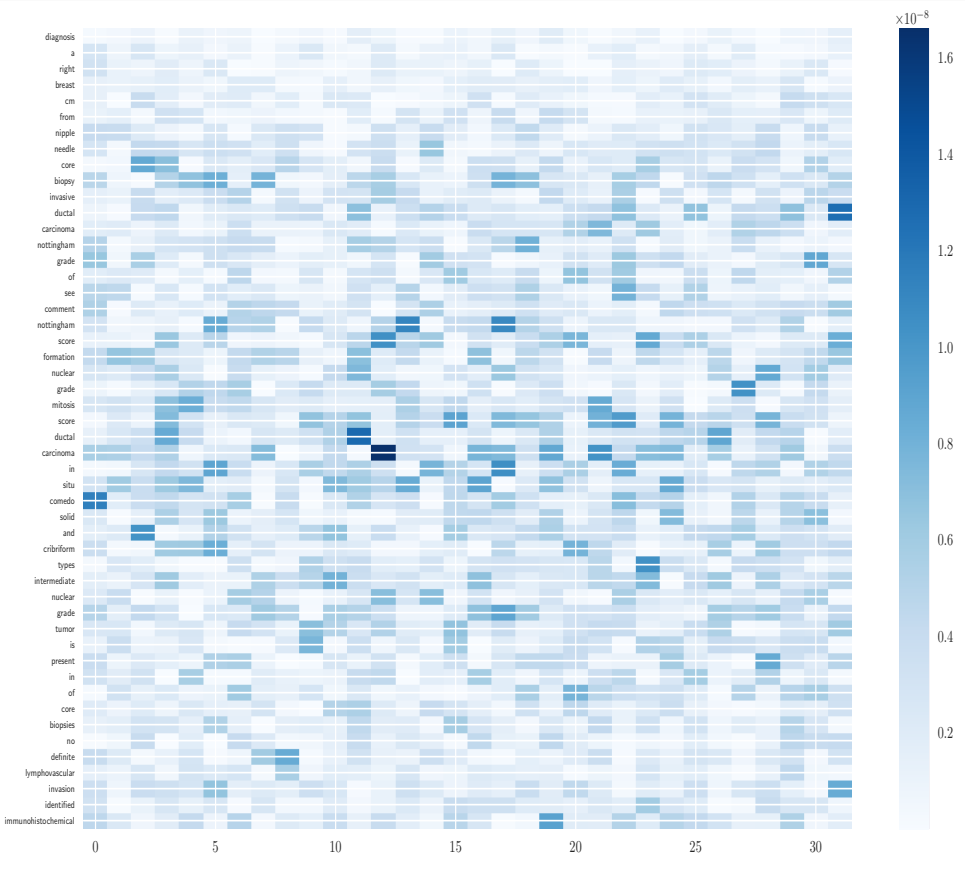
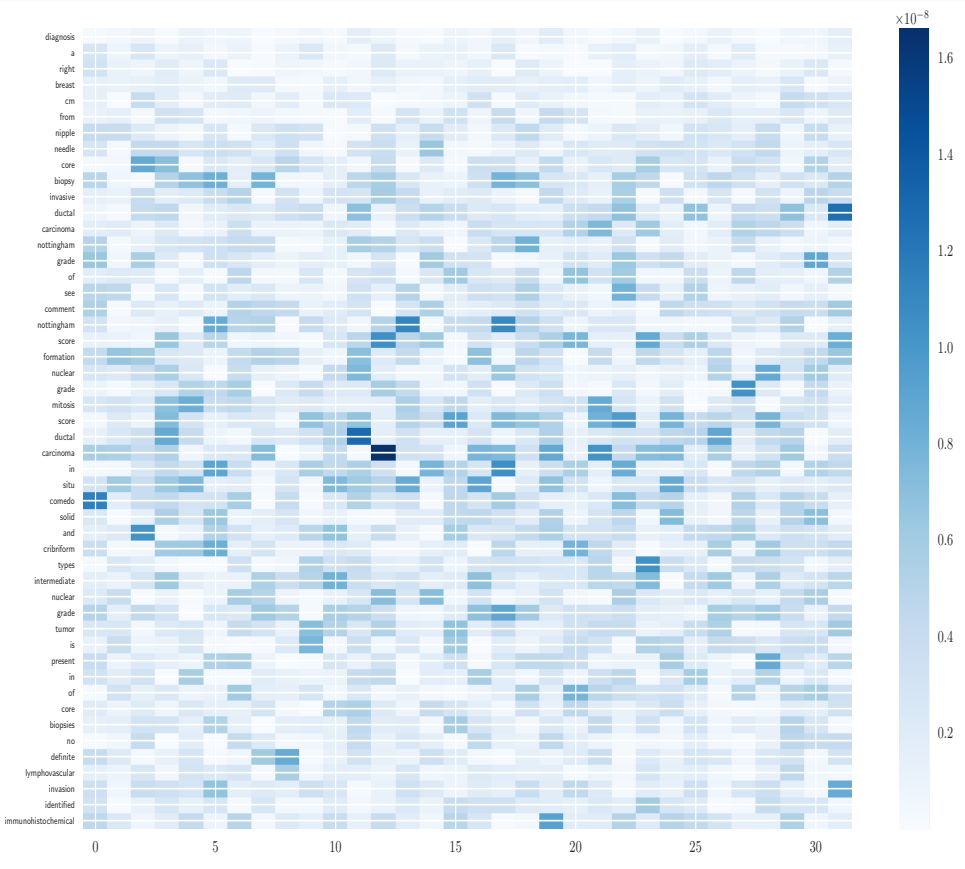


Figure 7: Visualize the importance of different tokens

Model Visualizations



- **carcinoma, ductal, biopsy, invasive, immunohistochemical, etc.**
- How confident are we with MT-CNN performance?

Figure 7: Visualize the importance of different tokens

Uncertainty Quantification

Uncertainty Quantification (UQ) of MT-CNN

Why UQ?

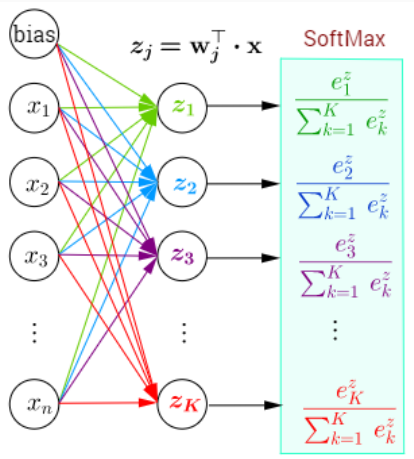
- Deep nets can confidently predict the wrong result
- Registries expect prediction confidence on a per report basis

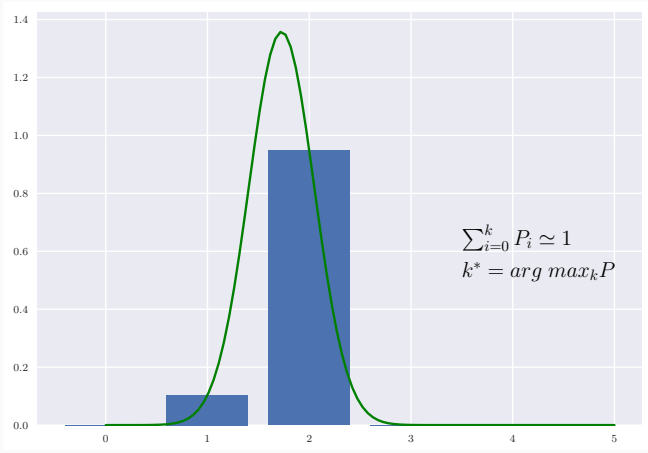
Uncertainty Quantification (UQ) of MT-CNN

Why UQ?

- Deep nets can confidently predict the wrong result
- Registries expect prediction confidence on a per report basis

- Each task of MT-CNN outputs a distribution, P, over all classes



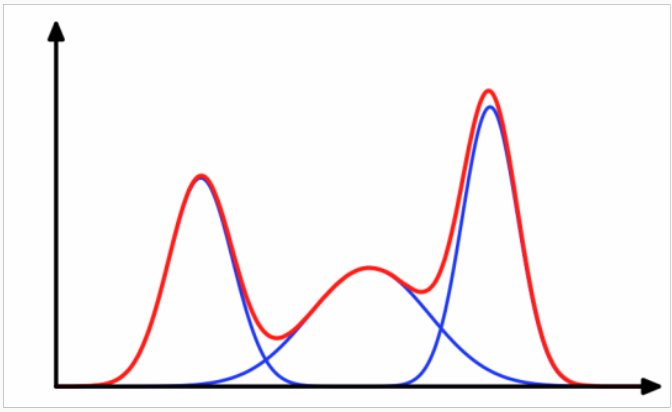


- These are not the actual probabilities, rather need to be calibrated against real-time performance to get true probabilities

  P = [0.01, 0.88, 0.04, 0.02, 0.05], k^* = **index(max(P)) = 2**

Mixture Models

- Presence of sub-populations with-in a sample population
- Use both *SoftMax* scores and *Entropy* to compute the confidence
- Formally, compute
  - $P(k_{true} = K | k^* = K, sofmax(P))$
  - $P(k_{true} = K | k^* = K, H(P))$



Mixture Model Formulation

Given population components π_1, π_2, ..., π_k and corresponding densities f_{π_1}, ..., f_{π_k} for a given population sample X, we find Confidence, $\mathscr{C}$,

$$\mathscr{C} = P(C(X) = \pi_k | X) = \frac{f_{\pi_k}(X) P(C(X) = \pi_k)}{\sum_j f_{\pi_j}(X) P(C(X) = \pi_j)}$$

C(X) denotes the component from which X was drawn.

UQ results

- Gaussian and Beta Mixture models on test data predictions

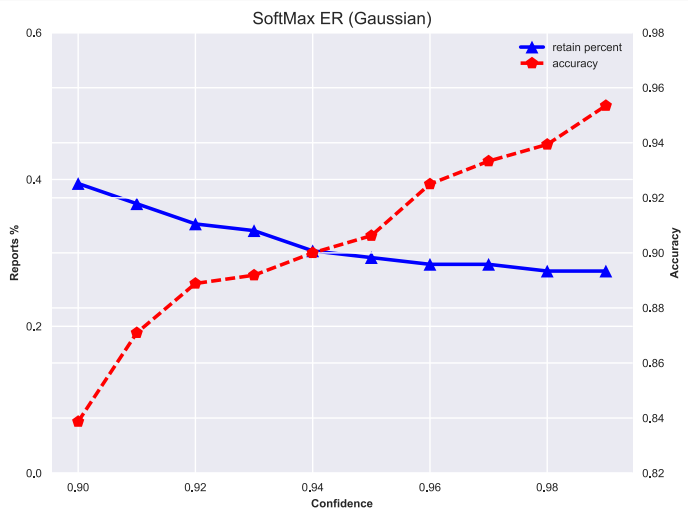


Figure 8: Gaussian UQ at $\mathscr{C} = 0.972$

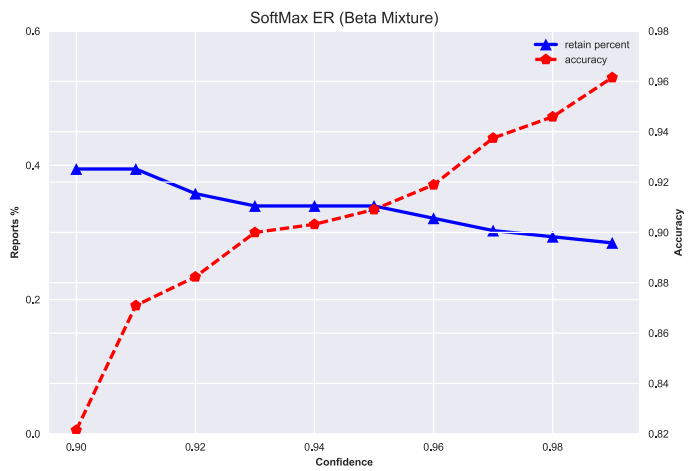
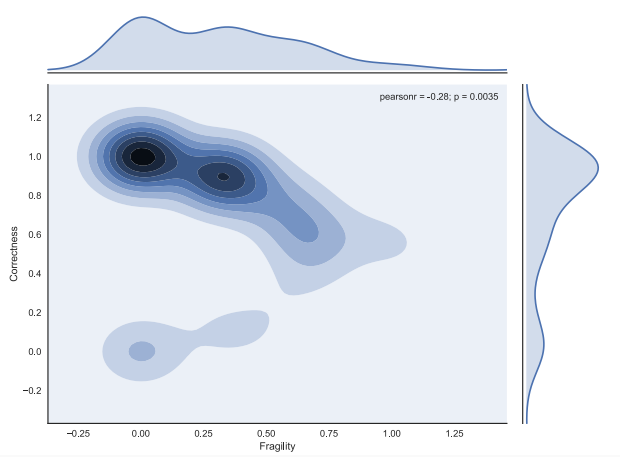


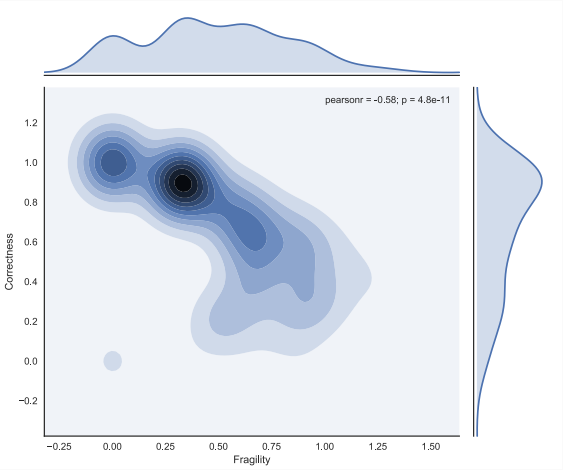
Figure 9: Beta Mixture Model UQ at $\mathscr{C} = 0.98$

- Retain Percent is the % of reports with $k^* \geq \mathscr{C}$
- Accuracy is the % of reports, where **predicted = true** and $k^* \geq \mathscr{C}$

Fragility vs. Correctness

- Fragility is the entropy of predictions over N number of experiments
- Correctness is the % of accurate predictions among N experiments.

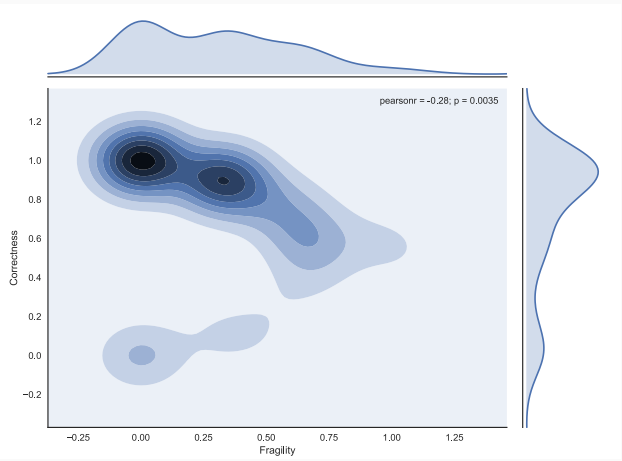


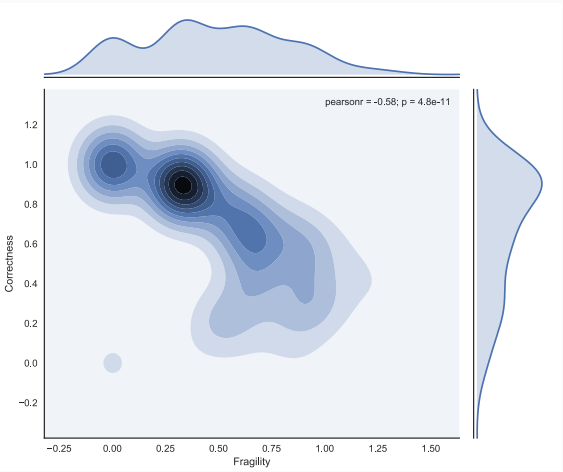


- 0.25 (left) and 0.5 (right) dropout rates

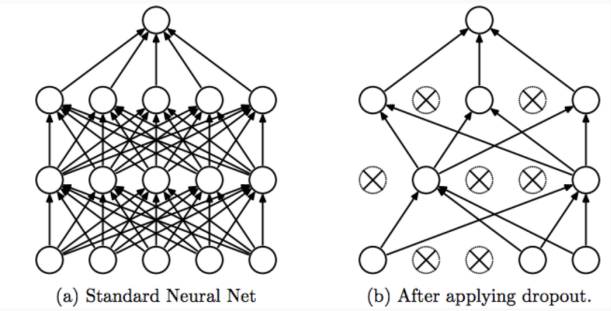
Fragility vs. Correctness

- Fragility is the entropy of predictions over N number of experiments
- Correctness is the % of accurate predictions among N experiments.





- 0.25 (left) and 0.5 (right) dropout rates


(a) Standard Neural Net (b) After applying dropout.

- More number of correct examples with lower entropy
- Perturbing the network has an impact on the performance.

Documenting the UQ results


```
{
 "task_name": "ER",    "report_index": 46,
 "Annotations": [ {
  "category": "ER",
  "classificationArray": [ {
    "classification": "Positive",
    "evidenceArray": [],
    "probability": {
        "est": 99.72696565410538,
        "raw_score": 0.9997360000000001,
        "calibrated": "yes",
        "est_type": "beta_mixtures"
  },
  "certainty": "yes"   } ]   } ]
}
```

Conclusion

Conclusion & Future Directions

- The MT-CNN approach is promising even after a rigorous UQ analysis.
- Confidence bounds are not enough, UQ is needed, especially on a per report basis confidence

Conclusion & Future Directions

- The MT-CNN approach is promising even after a rigorous UQ analysis.
- Confidence bounds are not enough, UQ is needed, especially on a per report basis confidence

Problem

However, there is always a need for labeled data in supervised classification.

Future Directions

We opt to employ a graph-based deep learning semi-supervised approach in order to predict the genetic markers.

Questions?