

Sequence Modelling of Collision Events with Convolutional Architectures

Justin Tan, Phillip Urquijo

2nd IML Workshop @ CERN

June 4, 2018



Motivation

Processes with a variable number of intermediate/final state particles occur in many contexts, e.g.

- Measurement of anomalous flavor observables
- Flavor tagging
- Vertexing
- Jet tagging

Traditionally, encode the event information in a fixed-dimensional vector as input to an ML algorithm, but this incurs some information loss.

Want a performant model that natively handles variable length sequences, while being competitive with current approaches.

Flavor Physics

Precision flavor physics

Compare precise experimental measurements of observables in B decays with theoretical predictions; interpret discrepancies in terms of new physics.

- Look for indirect effects of heavy unknown particles in low energy observables of B mesons.

Penguin processes:

Radiative: $b \rightarrow q\gamma$

Electroweak:

$b \rightarrow ql^+\ell^-$, $q = s, d$

- FCNCs, forbidden at leading order \rightarrow rare + hard to observe!

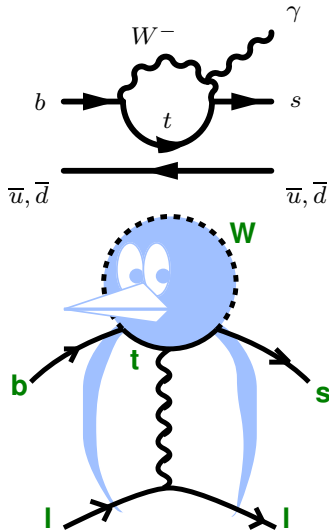


Figure 1: Radiative $b \rightarrow s\gamma$ (top) and electroweak $b \rightarrow sl^+\ell^-$ (bottom) penguins

Belle II

- Next generation B -physics experiment at SuperKEKB, an e^+e^- collider in Japan.
- Target: $50 \times 10^9 e^+e^- \rightarrow \Upsilon(4S) \rightarrow B\bar{B}$ events by 2024.
- Large statistics \rightarrow high precision measurements of penguin decay observables: $\mathcal{B}_{s(d)\gamma}, A_{CP}$.

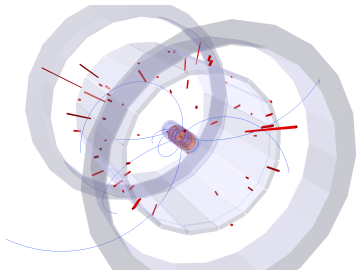


Figure 2: Belle II e^+e^- collision simulation.

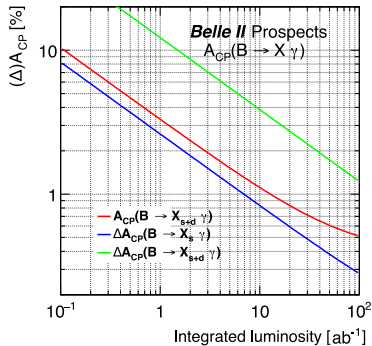


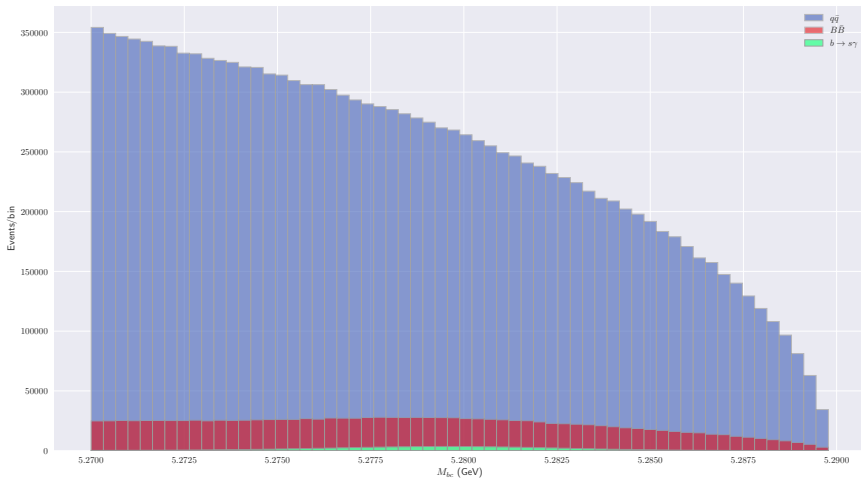
Figure 3: Sensitivity to A_{CP} (red) in $b \rightarrow s(d)\gamma$ decays.

Penguin hunting

Mass distribution for 1 ab^{-1} of simulated e^+e^- collisions at Belle II.

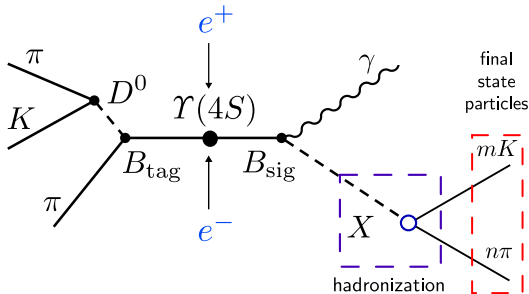
Background: $e^+e^- \rightarrow q\bar{q}$, $q = u, d, s, c$ + $e^+e^- \rightarrow b\bar{b}$

Signal: $b \rightarrow s\gamma$



Event Reconstruction

- Reconstruct $B_{\text{sig}} \rightarrow X\gamma$ from combining the radiative photon γ with the hadronic final state X
- Hadronic X is explicitly reconstructed in as many final states as possible (≈ 50)

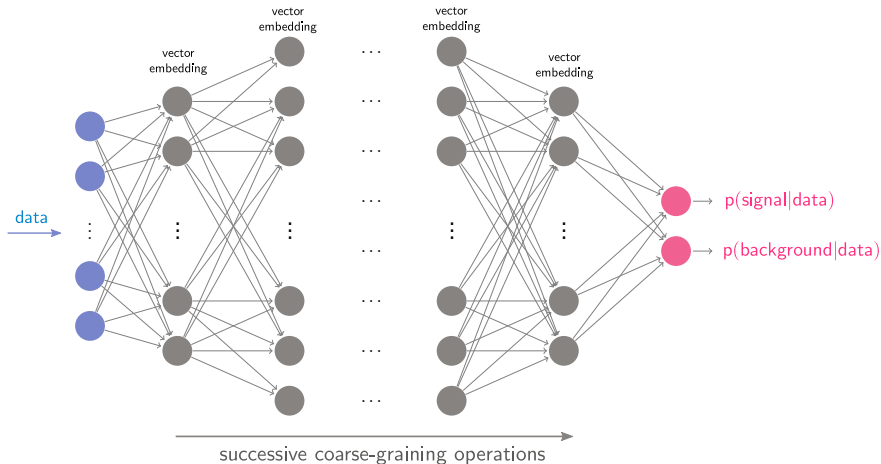


$$X = K^+\pi^+\pi^-$$
$$\text{or } (K_S^0 \rightarrow \pi^+\pi^-) \pi^+\pi^-$$
$$\text{or } \pi^+(\pi^0 \rightarrow \gamma\gamma)(\eta \rightarrow \gamma\gamma), \text{ etc.}$$

How can we capture the full event information of variable-length decay sequences?

Neural Networks

Identify 'relevant' degrees of freedom, iteratively integrate out 'irrelevant' degrees of freedom.



Natural Language

Condition on the entire sequence to infer a distribution over some property.

Prediction $p(\cdot | \text{Luke went to the beach and caught a})$

Translation $p(\text{some English} | \text{du français})$

Classification $p(\text{positive} | \text{Despite the constant negative press covfefe})$

Modern approach:

Introduce a learnable projection of each word into a continuous vector space, condition on the **learnt representation**, \mathcal{R} , usually the output of a neural network acting over the embedded words.



Collision event \leftrightarrow sentence, particle \leftrightarrow word

Particles are words in our 'language', described by a vector of 'morphemes':

$$\mathbf{x}_{particle} = (p^\mu, \{r, \theta, \phi\}) \leftarrow \text{kinematic} + \text{topological features}$$

e.g. Rare decay $B^+ \rightarrow \rho^+ \gamma$, where $\rho^+ \rightarrow \pi^+ \pi^0$ and $\pi^0 \rightarrow \gamma \gamma$, represent event as an ordered sequence of particle vectors

$$\{\mathbf{x}\}_{input} = \left[(\mathbf{x}_B, \mathbf{x}_\rho, \mathbf{x}_\gamma, \mathbf{x}_{\pi^+}, \mathbf{x}_{\pi^0}, \dots)^T \right]_{|p|-ordered}$$

Given the observed particle sequence, how probable is this correctly reconstructed signal?

$$p(\text{positive} \mid \text{Despite the constant negative press covfefe}) \ll 1$$

$$p(\text{signal} \mid \{\mathbf{x}_B, \mathbf{x}_\rho, \mathbf{x}_\gamma, \dots\}) = ?$$

Natural Language x Particle Physics

Decays can be very short ☺

$$B \rightarrow K^+ \pi^- \gamma, \quad \text{input: } \left[(\mathbf{x}_B, \mathbf{x}_{K^+}, \mathbf{x}_{\pi^-}, \mathbf{x}_\gamma)^T \right]_{|\mathbf{p}|\text{-ordered}}$$

Or very long ☺

$$B \rightarrow [K_S^0 \rightarrow \pi^+ \pi^-][\pi^0 \rightarrow \gamma\gamma][\pi^0 \rightarrow \gamma\gamma]\pi^+ \pi^- \gamma$$

$$\text{input: } \left[\left(\mathbf{x}_B, \mathbf{x}_{K_S^0}, \mathbf{x}_{\pi^+}, \mathbf{x}_{\pi^-}, \mathbf{x}_{\pi_{(1)}^0}, \mathbf{x}_{\pi_{(2)}^0}, \mathbf{x}_{\gamma_{(1)}^0}, \mathbf{x}_{\gamma_{(2)}^0}, \mathbf{x}_{\gamma_{(1)}^0}, \mathbf{x}_{\gamma_{(2)}^0}, \mathbf{x}_{\pi^+}, \mathbf{x}_{\pi^-}, \mathbf{x}_\gamma \right)^T \right]_{|\mathbf{p}|=o}$$

Challenging because of combinatorics for high-multiplicity states!

Instead of having a fixed ('global') representation of features present in all events, we use a variable-sized event representation to encode more information.

Event Representations

Classical representation

- Reliance on low-dimensional engineered features - information loss
- Can only use restrictive global event information - input is unordered set of features common to all event types
- No a priori knowledge of intrinsic structure of collision event

Sequential representation

- Inclusion of elementary kinematic features should contain all information needed to derive high-level features ✓
- Condition network response on all particle candidates in event → less information discarded ✓
- Introduce prior over event structure (composed of discrete units with related attributes) ✓

Two approaches to sequence analysis - *recurrent* and *convolutional*.

Recurrent models compress the entire history into a fixed-length vector, allowing long-range correlations to be understood.

- Read input sequence $X = (\mathbf{x}_1, \dots, \mathbf{x}_T)$
- Accumulate information in the hidden state $h = (\mathbf{h}_1, \dots, \mathbf{h}_T)$ through repeated matrix operations/nonlinearities

The **hidden state** h_n encodes knowledge about all particles encountered up to step n .

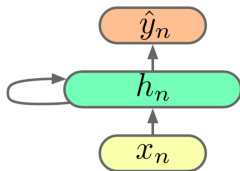


Figure 4: Network state factorizes into repeated application of hidden function \mathcal{H} .

Recurrency

The last hidden state h_T is a learnable encoding of the entire event.

- Input sequence: $X = (\mathbf{x}_1, \dots, \mathbf{x}_T)$
- Compute hidden vector sequence $h = (\mathbf{h}_1, \dots, \mathbf{h}_T)$

$$h_n = \mathcal{H}(V[x_n \oplus h_{n-1}] + b_h)$$

$$y_n = Wh_n + b_y, \quad |y_n| = \# \text{ classes}$$

$$p(c|X) = \text{softmax}(y_T), \quad \text{softmax}(v)_i = \frac{\exp v_i}{\sum_j \exp v_j}$$

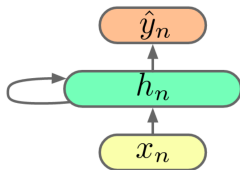
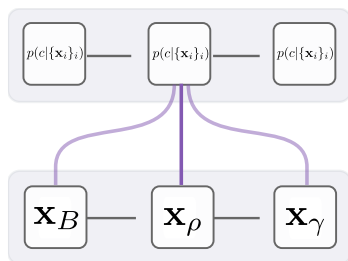
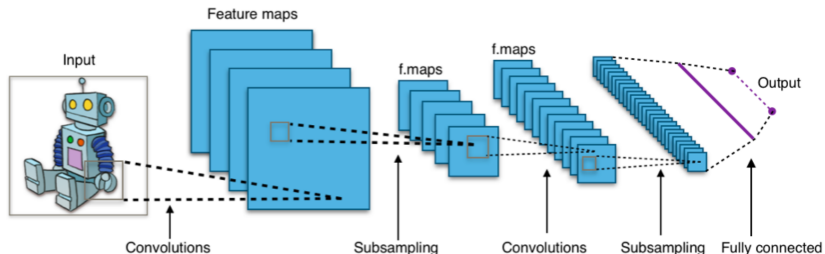


Figure 5: Network state factorizes into repeated application of hidden function \mathcal{H} .

- **Depth:** Multiple layers increases memory and representational capacity with linear computational increase
- **Bidirectionality:** Observe 'future' and 'past' context at each stage
- **Attention:** Impractical to encode all information about the sequence in a fixed size vector. Focus on subsets of information (different particles / features in the input collision) during prediction.
- **But:** Sequential operations cannot exploit parallelization ...



Convolutional Networks



Convolution: Capture local correlations between features in small regions of the input.

Subsampling: Coarse-graining: Extract important features from localized input regions.

Stacking convolutional layers: Build high-level features from first order local features → hierarchical feature development. Useful where local correlations in the input are crucial to prediction of global properties (e.g. computer vision, speech generation).

Image: [height, width, colors]

Collision: [1, # particles, features]

- Define filters which project raw features ('colors') into an embedding space to capture local correlations.
- Each filter runs over n adjacent particles simultaneously, projecting features from different particles into a common embedding space.
- Convolve filters across the input width, operating over each 'particle n -gram' (groups of n adjacent particles) in the sequence.
- Each filter is only aware of a local region of the input width, stack convolutional + subsampling layers to derive higher-order correlations/features

Convolutions

Input: Sequence $X = (\mathbf{x}_1, \dots, \mathbf{x}_T) \in \mathbb{R}^{1 \times T \times n_{\text{features}}}$

- Slide over particle n -grams with kernel $\mathbf{k}^{(n)} \in \mathbb{R}^{1 \times n \times K^{(n)}}$, with $K^{(n)}$ the embedding dimension, where $n = \{2 \dots 6\}$:

$$\mathbf{c}^{(n)} = (\mathbf{k}^{(n)} \star X) \in \mathbb{R}^{1 \times (T-n) \times K^{(n)}}$$

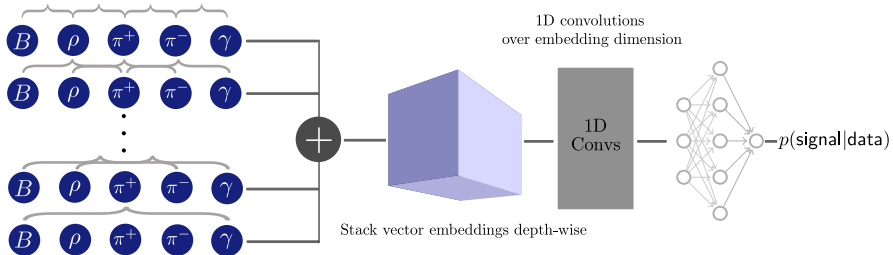
- Subsample (max/avg pool) over 2nd dimension to extract important features:

$$\mathbf{p}^{(n)} \in \mathbb{R}^{1 \times 1 \times K^{(n)}}$$

- Concatenate along first dimension: $\mathbf{f} = \text{concat}(\{\mathbf{p}^{(n)}\}_n, \text{axis}=1)$
- $\mathbf{f} \in \mathbb{R}^{1 \times j \times K}$, \leftarrow **stack of extracted feature maps**, $j = |\text{filters}|$
- Subject \mathbf{f} to further $[3, 1]$ convolutions to understand correlations between different feature maps, flatten + dense layer for final classification, using residual blocks here $\mathbf{x}_{n+1} = \mathbf{x}_n + \mathcal{F}(\mathbf{x}_n)$ gives small performance boost.

Convolve kernel $k^{(n)} \in \mathbb{R}^{n \times 1}$
 with particle n -grams
 $n = \{2, \dots, 6\}$

$$\gamma = (E_\gamma, \mathbf{p}_\gamma, \eta_\gamma, \dots)$$



Experiments

Run over simulated e^+e^- collisions at Belle II, with $\sqrt{s} = 10.58$ GeV.

Signal: Radiative penguin $b \rightarrow s\gamma$ ($B \rightarrow X_s\gamma$)

Background: $e^+e^- \rightarrow q\bar{q}$ and $e^+e^- \rightarrow B\bar{B}$, where both B mesons undergo non-penguin decay.

- Train: $\approx 23 \times 10^6$, test fraction 0.1
- Validation: $\approx 2 \times 10^6$

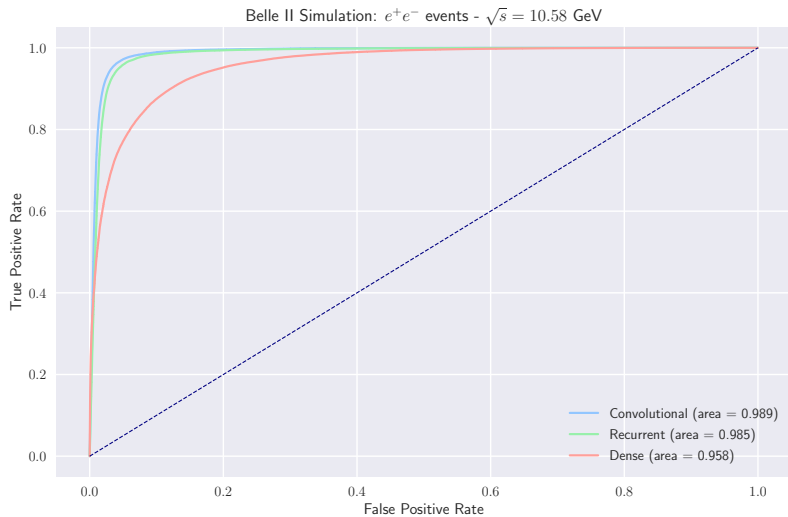
Represent the same events in two ways:

Fixed feature vector: Input to dense network

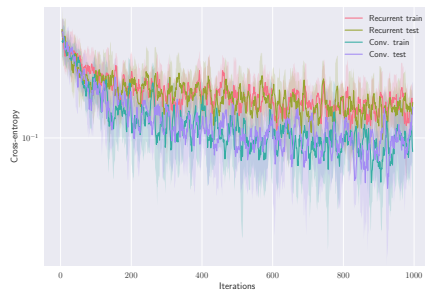
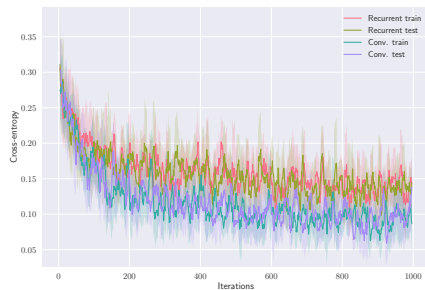
Variable-length sequence of vectors: Input to convolutional / recurrent nets

Recurrent architectures tend to overfit, but convolutional networks exhibit good generalization even with no explicit regularization.

Results

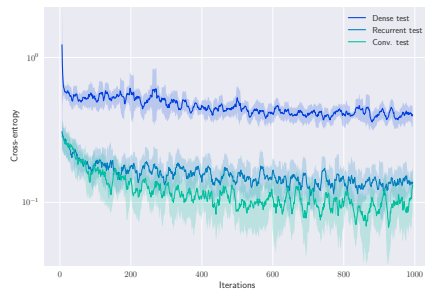
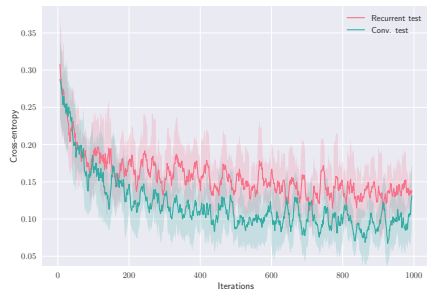


Convergence



Kernel weight sharing in convolutional networks has a strong regularizing effect.

Convergence



Performance

Learnable parameters

Kernel weight sharing in CNNs + parameter-less pooling layers → reduced learnable parameters relative to recurrent/dense networks.

Computation Time

Recurrent architectures perform sequential computation → unable to exploit the parallelization capabilities of modern GPUs.

TensorFlow 1.7 | CUDA 9.0 | 2 Tesla P100s | $b \rightarrow s\gamma$

Architecture	AUC [†]	$\frac{\text{Training time}}{\text{Conv. training time}}$	Learnable parameters
Convolutional	0.988 ± 0.03	1	$\approx 4.5 \times 10^5$
Recurrent	0.985 ± 0.04	3.9	$\approx 1.2 \times 10^6$
Dense*	0.956 ± 0.04	0.6	$\approx 2.3 \times 10^6$

*Same events, but cast in sequential representation for conv./recurrent models - on average lower # features/event used for dense network.

[†]Training rerun 5 times with different random seeds for weight initialization

Based on EWP skim cuts + additional tight criteria

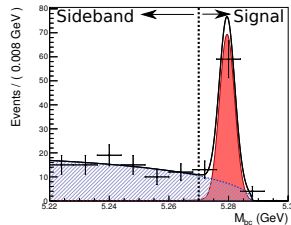
Decay	Architecture	AUC	$R_{\epsilon_s=0.5}$
$b \rightarrow s\gamma$	Convolutional	0.988 ± 0.03	-
	Recurrent	0.985 ± 0.04	-
	Dense	0.956 ± 0.04	-
$b \rightarrow sll$	Convolutional	0.995 ± 0.02	-
	Recurrent	0.986 ± 0.02	-
	Dense	0.934 ± 0.01	-

Table 1: Shown is the mean of 8 experiments with different random seeds for the initial network weights. Hyperparameters are tuned independently based on the same computational budget.

$b \rightarrow c\tau\bar{\nu}$ coming soon ...

Signal Identification

- Identify signal peak in:
 - ▶ $\Delta E = E_{beam} - E_B$
 - ▶ $M_{bc} = \sqrt{E_{beam}^2 - |\vec{p}_B|^2}$
- Background processes not fully captured by simulation
- Rely on interpolation of smooth background spectrum from sidebands beneath signal peak



Learning algorithms preferentially select signal-like events → background spectrum distortion → **uncontrollable systematic uncertainties**

Background Sculpting

Classifier output $f(X; \theta_f) \sim p(\text{signal}|\text{data})$. Only accept events above a given posterior probability.

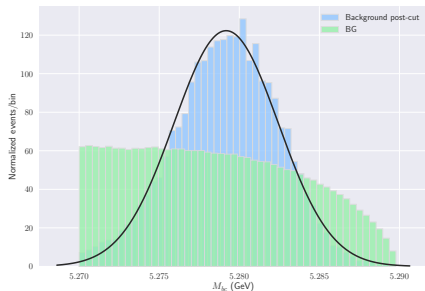


Figure 6: Continuum M_{bc} before (green) and after (blue) 0.995 suppression

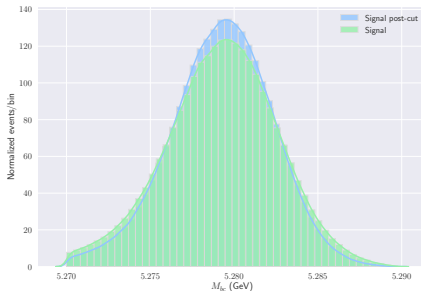
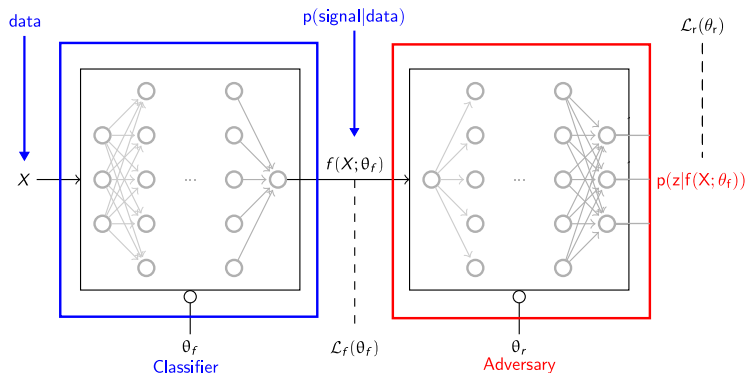


Figure 7: Signal M_{bc} before (green) and after (blue) 0.995 suppression

Background looks like signal post-selection.

Tension between optimal discrimination and reduced systematics!



Adversary r attempts to infer z from $p(\text{signal}|\text{data})$ emitted by the classifier f , increasing the loss function $E = \mathcal{L}_f(\theta_f) - \mathcal{L}_r(\theta_r)$.

f circumvents penalization by decorrelating $p(\text{signal}|\text{data})$ with z .

Fisher Information

Measure of amount of information that an observed variable X holds about a hidden variable θ

- X : Network output
- θ : 'Fit variables'

Given a model $p_\theta(x)$ that yields a log likelihood $\log p_\theta(x)$, the Fisher Information measures the curvature of the LL:

$$\begin{aligned} F(\theta) &= \mathbb{E}_X \left[\left(\frac{\partial}{\partial \theta} \log p_\theta(x) \right)^2 \middle| \theta \right] \\ &= -\mathbb{E}_X \left[\frac{\partial^2}{\partial \theta^2} \log p_\theta(x) \middle| \theta \right] \end{aligned}$$

Fisher Information

Enforce decorrelation of network output X with θ by penalizing the magnitude of the Fisher Information.

$$\mathcal{L}(\psi_f; \theta) = \mathbb{E}_X [-\log p_{\psi_f; \theta}(x)] + \lambda_F F(\theta) \quad (1)$$

For multiple 'fit-variables': $\theta = (\theta_1, \dots, \theta_N)^T$

$$F(\theta)_{ij} = \mathbb{E}_X \left[\left(\frac{\partial}{\partial \theta_i} \log p_\theta(x) \right) \left(\frac{\partial}{\partial \theta_j} \log p_\theta(x) \right) \middle| \theta \right]$$
$$\mathcal{L}(\psi_f; \theta) = \mathbb{E}_X [-\log p_{\psi_f; \theta}(x)] + \lambda_F \text{Tr } F(\theta) \quad (2)$$

Fisher Information - Experiments

- Signal classification in the rare flavor-changing neutral current transition $b \rightarrow s\ell\ell$, want to enforce decorrelation with $\theta = M_{bc} \equiv \sqrt{E_{beam}^2 - |\vec{p}_B|^2}$.
- Simple architecture: 5 layer densely connected network with 512 nodes per layer, SGD ($\beta = 0.9$) + Nesterov Momentum ($\gamma = 0.9$).
- 6M training events, 6×10^5 test
- $\lambda_F = 100$
- Very easy to isolate in M_{bc} !
- Use empirical estimate of Fisher

$$F(\theta) \approx \frac{1}{N} \sum_{n=0}^N \left(\frac{\partial}{\partial \theta} \log p_{\theta}(x_n) \right)^2 \quad (3)$$

Fisher Information - No Penalty

Fisher Information - Experiments

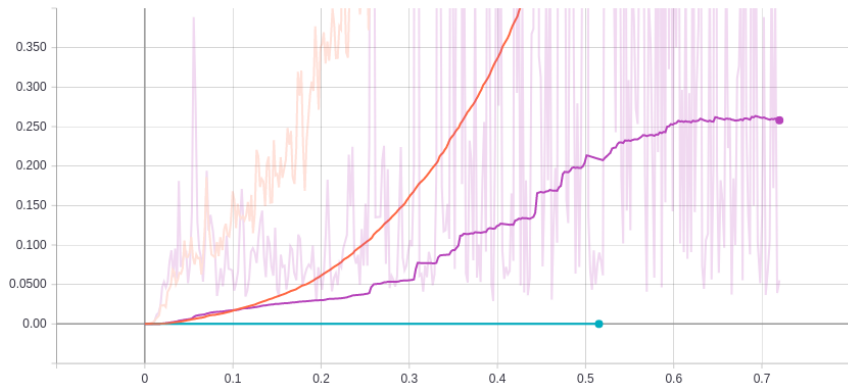
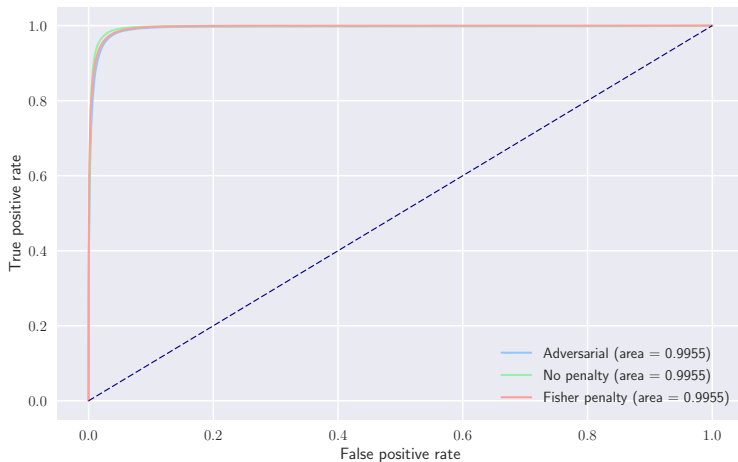


Figure 8: Fisher penalty (blue), no penalty (orange) and adversary (purple)

Fisher Information - Experiments

Receiver operating characteristic

$b \rightarrow sll$, Decorrelation comparison



Fisher Information

- **Simplicity:** single hyperparameter λ_F , adversarial method roughly doubles the number of hyperparameters ($\lambda_{\text{adv.}}$, adversary learning rate, relative iterations, adversary architecture, etc.), e.g. grid search with fineness 10 scales like $\sim 10^{2N}$ compared to 10^{N+1}
- Easy to accommodate multiple nuisance parameters
- Convergence comparatively easier to achieve compared to adversarial method
- **However:** independence constraint not as strong as adversarial method due to correlations between input variables and nuisance parameters.
- Only works with simple dense architectures at the moment.

Summary

- Draw parallels between event structure / natural language
- Represent a collision event as an ordered set of feature vectors, one for each reconstructed particle candidate
 - ▶ Capture more complete picture of event than classical approaches
- Convolutional architectures permit sequential event representation
 - ▶ Capture local interactions between particle candidates through convolution + subsampling
 - ▶ Long-range / global relations can be understood by stacking convolutional layers → increase in receptive field
- Why convolutions?
 - ▶ Fast! Exploits parallelization, unlike recurrent approaches
 - ▶ Outperforms recurrent/dense approaches

Improved background rejection → better sensitivity to new physics.

Thanks for listening

Working on a formal writeup, with applications to:

- $b \rightarrow s(d)\gamma$, $b \rightarrow sll$
- $b \rightarrow cl\nu$
- Vertex reconstruction

Plus: investigating methods to encode information about particle identity and mother/daughter relationships into each particle feature vector, Glove-like embeddings.

Code + Docs

Github: github.com/Justin-Tan/particle2seq

Stash: stash.desy.de/users/jtan/repos/event2seq

justin.tan@coepp.org.au

Backup



Implementation

- ▶ Data collection: ROOT
- ▶ To Python: uproot
- ▶ Preprocessing: Spark/Pandas
- Workflow scalable to $\mathcal{O}(100)$ GB worth of training data.
- TensorFlow:
 - ▶ Open-source: No black boxes. ✓
 - ▶ Fine-grained control over entire architecture. ✓
- Train:
 - ▶ 64 epochs, scheduled annealing
 - ▶ SGD + Nesterov momentum



Motivation

- Non-SM contributions enter through hypothetical new TeV-scale particles running within the loop \rightarrow interference with known amplitudes.
- Strong constraints on NP by measurement of inclusive/exclusive BR, CP asymmetries

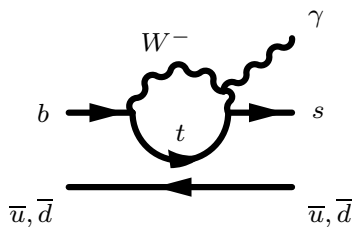


Figure 10: Example of SM radiative penguin decay for $b \rightarrow s\gamma$ [2]

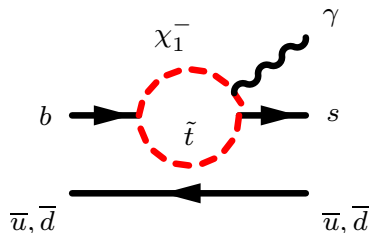


Figure 11: Example of hypothetical SUSY contribution to radiative decay [2]

Words as Vectors

Distributional Hypothesis: Words that occur in the same context share semantic meaning.

- Represent words in a continuous vector space to group semantically similar words.
 - ▶ Learned vectors explicitly encode linguistic regularities and patterns:
 $\vec{v}(\text{Madrid}) - \vec{v}(\text{Spain}) + \vec{v}(\text{France}) \approx \vec{v}(\text{Paris})$
 - ▶ Inability to represent idiomatic phrases
 $\vec{v}(\text{California}) \neq \vec{v}(\text{Golden}) + \vec{v}(\text{State})$ - overcome with phrase based models.

Takeaway: Encode semantic relationships in directions in induced vector space.

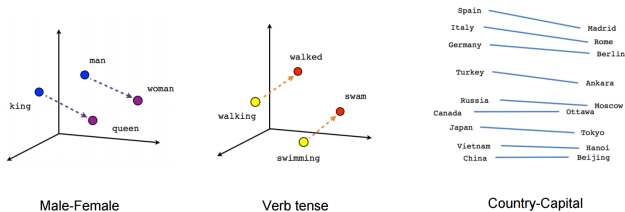


Figure 12: Semantic relationships as approximate linear relations (projected into 3D)

Stacking Recurrent Layers

A Deep RNN increases memory and representational capacity with linear scaling.

- The output sequence of one layer forms the input sequence for the next

$$h_t^{(n)} = \mathcal{H} \left(V^{(n)} \left[h_t^{(n-1)} \oplus h_{t-1}^{(n)} \right] + b_h^{(n)} \right)$$

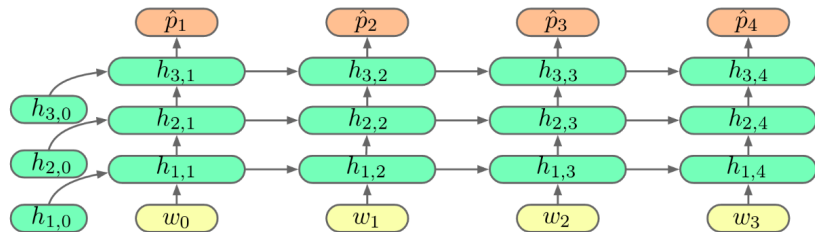
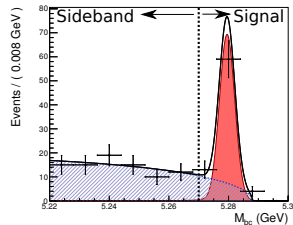


Figure 13: Hidden state of layer n accepts hidden state of layer $n - 1$ as input [5]

Signal Identification

- Identify signal peak in:
 - ▶ $\Delta E = E_{beam} - E_B$
 - ▶ $M_{bc} = \sqrt{E_{beam}^2 - |\vec{p}_B|^2}$
- Background processes not fully captured by simulation
- Rely on interpolation of smooth background spectrum from sidebands beneath signal peak



Learning algorithms preferentially select signal-like events → background spectrum distortion → **uncontrollable systematic uncertainties**

Background Sculpting

Classifier output $f(X; \theta_f) \sim p(\text{signal}|\text{data})$. Only accept events above a given posterior probability.

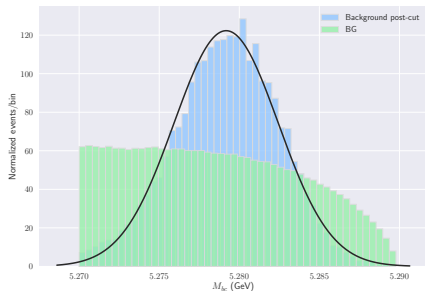


Figure 14: Continuum M_{bc} before (green) and after (blue) 0.995 suppression

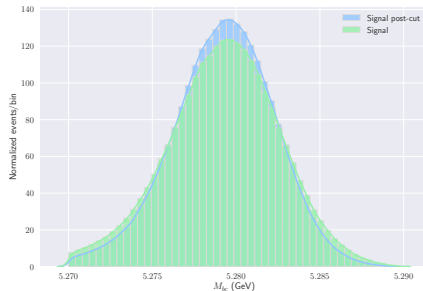


Figure 15: Signal M_{bc} before (green) and after (blue) 0.995 suppression

Background looks like signal post-selection.

Tension between optimal discrimination and reduced systematics!

Controlling Systematics

- Physics variables of interest: $z \in \mathcal{Z}$ (e.g. $\Delta E, M_{inv}$)
- Classification function: $f(X; \theta_f)$ gives probabilities of data X being signal events.
- $f(X; \theta_f)$ and z should be independent random variables

$$p(f(X; \theta_f) = s | z) = p(f(X; \theta_f) = s | z')$$

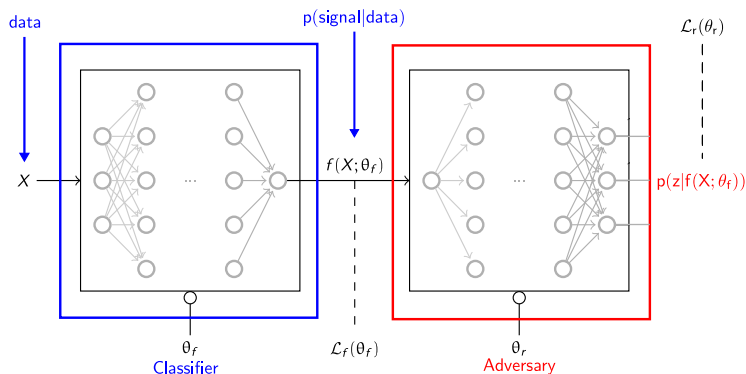
Q: How can we enforce independence of $f(X; \theta_f)$ and Z ?

A: Set up a game between two competing players, f and r . Independence arises at the Nash equilibrium.

Train f and r simultaneously by minimax optimization of

$$\hat{\theta}_f, \hat{\theta}_r = \arg \min_{\theta_f} \left(\max_{\theta_r} (\mathcal{L}_f(\theta_f) - \mathcal{L}_r(\theta_r)) \right)$$

Adversarial Neural Networks



Adversary r attempts to infer z from $p(\text{signal}|\text{data})$ emitted by the classifier f , increasing the loss function $E = \mathcal{L}_f(\theta_f) - \mathcal{L}_r(\theta_r)$.

f circumvents penalization by decorrelating $p(\text{signal}|\text{data})$ with z .

Adversarial Neural Networks

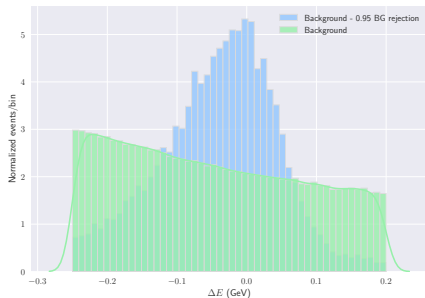


Figure 16: Standard neural network

- Enforce 95% BG rejection
- Signal: $b \rightarrow s\gamma$
- Background: $e^+e^- \rightarrow q\bar{q}$

Smooth interpolation from sideband ✓

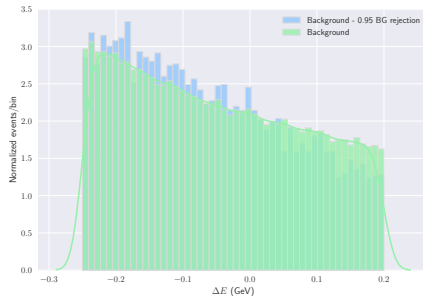


Figure 17: Adv. trained neural network

Adversarial Neural Networks

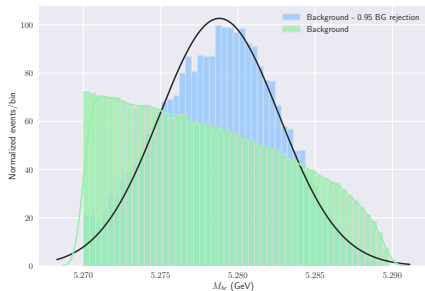


Figure 18: Standard neural network

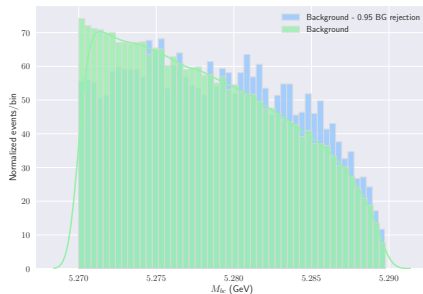


Figure 19: Adv. trained neural network

- Enforce 95% BG rejection on 1 ab^{-1} of simulated e^+e^- collisions at Belle II
- Signal: $b \rightarrow s\gamma$
- Background: $e^+e^- \rightarrow q\bar{q}$

Smooth interpolation from sideband ✓

No Free Lunch

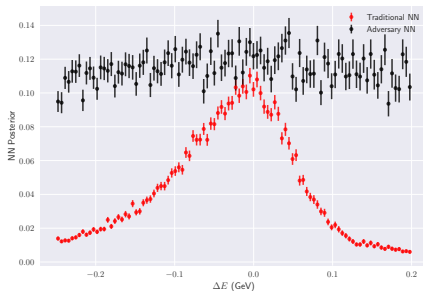


Figure 20: Posterior $p(\text{signal}|\text{data})$ versus ΔE

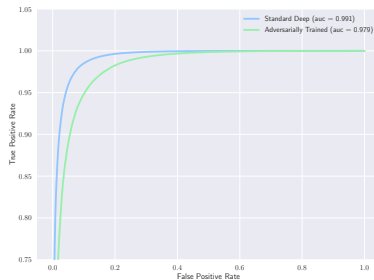


Figure 21: Competition reduces separation power

- Posterior probabilities relatively uniform ✓
- Tradeoff between optimal discrimination and reduced systematic error. ✗

Fisher Information - Adversarial

Fisher Information - Fisher Penalty