Enhanced Higgs to $\tau^+\tau^-$ Search with Deep Learning

P. Baldi and P. Sadowski
Dept. of Computer Science, UC Irvine, Irvine, CA 92617

D. Whiteson
Dept. of Physics and Astronomy, UC Irvine, Irvine, CA 92617

The Higgs boson is thought to provide the interaction that imparts mass to the fundamental fermions, but while measurements at the Large Hadron Collider (LHC) are consistent with this hypothesis, current analysis techniques lack the statistical power to cross the traditional 5σ significance barrier without more data. *Deep learning* techniques have the potential to increase the statistical power of this analysis by *automatically* learning complex, high-level data representations. In this work, deep neural networks are used to detect the decay of the Higgs to a pair of tau leptons. A Bayesian optimization algorithm is used to tune the network architecture and training algorithm hyperparameters, resulting in a deep network of eight non-linear processing layers that improves upon the performance of shallow classifiers even without the use of features specifically engineered by physicists for this application. The improvement in discovery significance is equivalent to an increase in the accumulated dataset of 25%.

INTRODUCTION

Observations made at the LHC led to the announcement of the discovery of the Higgs boson in 2012 [1, 2], and much more data will be collected when the collider comes back online in 2015. A top priority is to demonstrate that the Higgs boson couples to fermions through direct decay modes. Of the available modes, the most promising is the decay to a pair of tau leptons ($\tau^\pm$), which balances a modest branching ratio with manageable backgrounds. From the measurements collected in 2011-2012, the LHC collaborations report data consistent with $H \to \tau^+\tau^-$ decays, but without statistical power to cross the 5σ threshold, the standard for claims of discovery in high-energy physics. There is a vigorous effort in the high-energy physics community to improve the statistical analysis of collider data in order to require smaller accumumated datasets for this scientific discovery and others like it.

Machine learning is already widely used in the $H \to \tau^+\tau^-$ search and other areas of high-energy physics, but standard software packages primarily rely on shallow learning models such as artificial neural networks with only a single hidden layer. Recent interest in *deep learning* has resulted in significant advances in computer vision and speech recognition. This technique uses deep neural networks with multiple non-linear hidden layers, which are able to represent complex functions more efficiently than shallow networks, and may generalize better to new data due to architectural constraints [3]. While training such networks is notoriously difficult due to the vanishing gradient problem [4, 5], recent advances in computing hardware have made it feasible to train deeper networks on larger datasets.

In the field of high-energy physics, deep neural networks have demonstrated an ability to significantly increase classification performance and discovery significance on two other high-energy physics applications [6]. This is important, as the colliders are expensive to operate and the particle detection systems have short lifetimes due to the intense radiation produced in collisions. Therefore, boosting the discovery significance can shorten the time needed to make a discovery, or make discoveries possible in limited-size datasets. Because classifiers can be trained on an arbitrary quantity of simulated data, large neural network architectures with millions of parameters can be trained without overfitting. Thus, the challenge lies in selecting an appropriate set of hyperparameters that determine the network architecture and training algorithm details that yield the best-fitting classifier.

In this Letter, we apply deep learning techniques to the important application of detecting $H \to \tau^+\tau^-$ decays, where it is worth the computational cost to tune these parameters carefully in order to maximize the statistical power of the analysis. The hyperparameters are optimized systematically using a Bayesian optimization algorithm.

MODEL

Proton collisions at the LHC annhiliate the proton constituents, quarks, and gluons. In a small fraction of these collisions, a new heavy state of matter forms, such as a Higgs or Z boson. These heavy states are unstable, quickly decaying into successively lighter and more stable particles. In the case of a Higgs boson, the decay process is:

$$gg \to H \to \tau^+\tau^- \qquad (1)$$

In the study presented here, the decay of τ leptons into lighter leptons (e and μ) and pairs of neutrinos (ν),

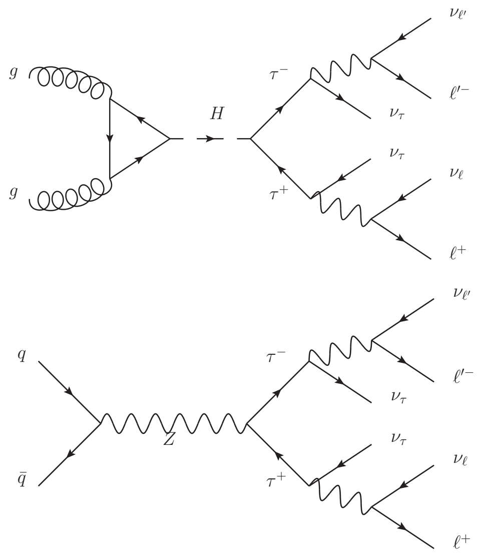

FIG. 1. Feynman diagrams describing the signal $gg \to H \to \tau^+\tau^- \to \ell^-\nu\nu\ell^+\nu\nu$ process and the dominant background $q\bar{q} \to Z \to \tau^+\tau^- \to \ell^-\nu\nu\ell^+\nu\nu$ process.

$\tau^\pm \to \ell^\pm\nu_\tau\nu_\ell$ is considered; see Fig. 1. This is the most challenging decay mode, as it involves the largest number of invisible neutrinos.

Detectors surrounding the point of collision measure the identity, momentum, and direction of the visible final stable particles; the intermediate states of matter are not observable. Two processes that generate the same sets of stable particles can be difficult to distinguish. Figure 1 shows how the process $q\bar{q} \to Z \to \tau^+\tau^-$ yields the identical list of particles as a process that produces the Higgs boson.

To distinguish between two processes with identical final state particles, the momentum and direction of the visible final state particles are examined closely. Given perfect measurement resolution and a complete description of the final state particles B and C, the invariant mass of the short-lived intermediate state A in the process $A \to B + C$ is given by:

$$m_A^2 = m_{B+C}^2 = (E_B + E_C)^2 - |(p_B + p_C)|^2 \qquad (2)$$

However, finite measurement resolution and escaping neutrinos (which are invisible to the detectors) make it impossible to calculate the intermediate state mass precisely. Instead, the momentum and direction of the final state particles are studied. Sophisticated Monte Carlo programs have been carefully tuned to produce highly faithful collision simulations, and simulated data is used to investigate methods for distinguishing between possible generating processes. In the studies shown here, all samples are generated with the MADGRAPH5 [7] program, with showering and hadronization performed by PYTHIA [8] and ATLAS detector simulation with DELPHES [9]. Machine learning can then be applied to learn a classification model from this simulated data. These classifiers take as input the variables that are measured by the detectors (and/or high-level variables that are derived from these measurements), and learn to predict the probability that a given example was the result of a particular generating process. The relevant variables for $H \to \tau^+\tau^-$ classification are described below.

Low-level variables

Ten essential measurements are provided by the detectors:

- The three-dimensional momenta, p, of the charged leptons;
- The imbalance of transverse momentum ($\not{E}_T$) in the final state transverse to the beam direction, due to unobserved or mismeasured particles;
- The number and momenta of particle 'jets' due to radiation of gluons or quarks.

Distributions of these variables in simulation are given in Fig. 2.

High-level variables

In order to better discriminate between Higgs-boson production and Z-boson production, there is a vigorous effort to construct non-linear combinations of these low-level variables that capture useful high-level information. The derived variables that have been considered include:

- Axial missing momentum, $\not{E}_T \cdot p_{\ell^+\ell^-}$;
- Scalar sum of the observed momenta, $|p_{\ell^+}| + |p_{\ell^-}| + |\not{E}_T| + \sum_i |p_{\text{jet}_i}|$;
- Relative missing momentum, $\not{E}_T$ if $\Delta\phi(p, \not{E}_T) \geq \pi/2$, and $\not{E}_T \times \sin(\Delta\phi(p, \not{E}_T)$ if $\Delta\phi(p, \not{E}_T) < \pi/2$, where p is the momentum of any charged lepton or jet;
- Difference in lepton azimuthal angles, $\Delta\phi(\ell^+, \ell^-)$;
- Difference in lepton polar angles, $\Delta\eta(\ell^+, \ell^-)$;
- Angular distance between leptons, $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$;
- Invariant mass of the two leptons, $m_{\ell^+\ell^-}$;
- Missing mass, m_{MMC} [10];
- Sphericity and transverse sphericity;

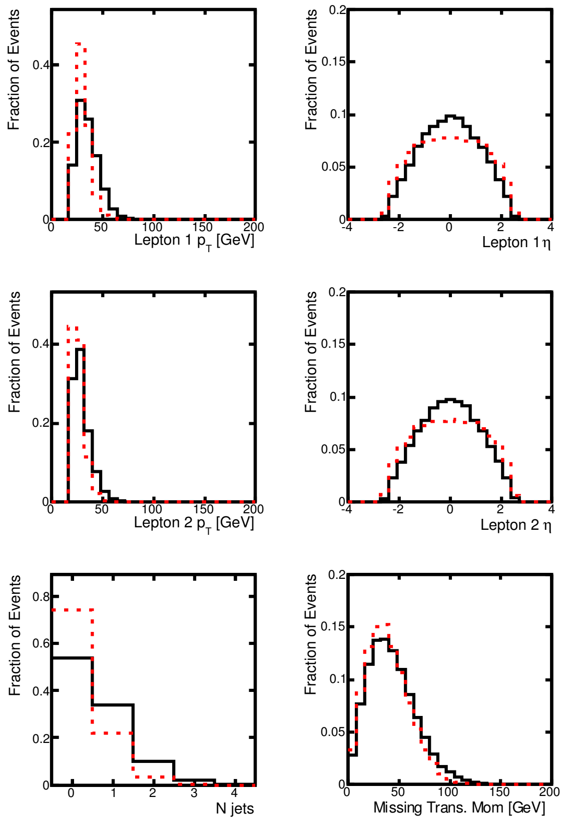

FIG. 2. Distributions of low-level input variables from basic kinematic quantities in $\ell\ell + \not\!\!E_T$ events for simulated signal (black, solid) and background (red, dashed) benchmark events. Shown are the distributions of transverse momenta (p_{T}) and azimuthal angle (η) of each observed particle as well the number of hadronic jets (N_{jets}) and the imbalance of transverse momentum ($\not\!\!E_T$) in the final state. Polar angle (ϕ) information for each observed particle is also available to the network, but is not shown, as the one-dimensional projections have little information.

- Invariant mass of all visible objects (leptons and jets).

Distributions of these variables in simulation are given in Fig. 3. At first glance, these high-level variables appear to contain more discriminatory power than the low-level variables.

RESULTS

Deep neural networks were trained with state of the art techniques to detect $H \to \tau^+\tau^-$ decay events using a Bayesian optimization algorithm to select hyperparameters. The results are compared to a similar optimization over shallow neural networks with the same number of tunable model parameters, as well as networks trained on the two feature types: low-level variables and high-

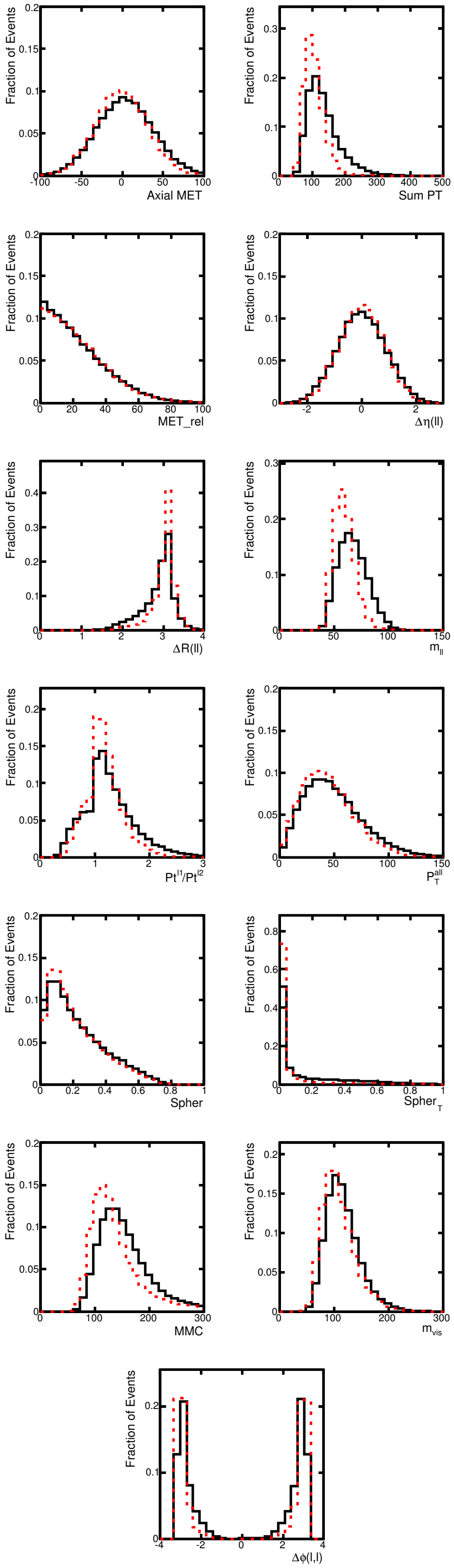

FIG. 3. Distributions of high-level input variables derived from the low-level variables in $\ell\ell + \not\!\!E_T$ events for simulated signal (black, solid) and background (red, dashed) benchmark events. See text for definitions.

level variables.

Hyperparameters for the deep neural network architecture and training algorithm were selected to minimize the expected generalization loss using the Spearmint Bayesian optimization algorithm [11, 12]. The algorithm was allowed to perform 100 experiments; each experiment tested a unique combination of hyperparameters, training a neural network on a training data set of 40 million random examples and computing the generalization error on another 10 million random examples. The following hyperparameters were optimized simultaneously: the learning rate decay factor, the initial momentum, the final momentum, the number of epochs until momentum saturation, the number of hidden layers, and the number of neurons per hidden layer. All hidden layers were constrained to have the same number of neurons. Additional details regarding the hyperparameter search space can be found in the Methods section.

The best hyperparameter combination used the maximum of eight layers. Training was repeated five times using the optimized hyperparameters, random train/test splits, and random weight initializations. Table I shows the mean and variance of two performance metrics; the deep neural networks (DNN) achieve a mean Area Under the Receiver Operator Characteristic Curve (AUC) of 0.802 (s.d. 0.0001), and the corresponding expected significance of discovery is 3.37 (s.d. 0.003) Gaussian σ. The expected significance is calculated using $N_{\text{sig}} = 100, N_{\text{backg.}} = 5000 \pm 250$ with a profile likelihood method and evaluated with the asymptotic approximation [13]. An additional performance boost is obtained by creating an ensemble classifier from the five networks, which achieves an AUC of 0.803 and discovery significance of 3.39σ. Figure 4 shows the effect of network depth on performance.

For comparison, we performed the same Bayesian optimization on a set of shallow neural networks (NN). The search space included shallow networks with up to 56,000 hidden units, which have the same number of tunable model parameters as the largest deep networks. However, a single, large, hidden layer did not lead to better performance; the best network had just 691 hidden units. These shallow architectures performed significantly worse than the deep networks, even when using an ensemble (Table I, Figure 5).

The contribution of the high-level variables derived by physicists was analyzed by training on the different feature subsets. Deep and shallow neural networks were trained on both the 10 low-level variables and the 15 high-level variables only, using the same hyperparameters that had been optimized for the networks trained on the full feature set. Table I shows that the networks perform better with the high-level variables, but that they perform best with the complete set. To put into practical context the impact of the boost in discovery significance between the NN and the DNN, we measure the increase needed in the size of the expected data set (nominally $N_{\text{sig}} = 100, N_{\text{backg.}} = 5000 \pm 250$) to achieve the same enhancement in discovery significance ($3.02\sigma \rightarrow 3.37\sigma$) for the NN; an enlargement of 25% is required. Therefore, using the DNN dramatically shortens the time needed to operate the collider before the data are statistically significant.

TABLE I. Comparison of the performance of shallow neural networks (NN), and deep neural networks (DNN) for three sets of input features: low-level variables, high-level variables, and the complete set of variables. Each neural network was trained five times with different random weight initializations and different train/test splits. Performance of an ensemble classifier using all five instances is also shown. The table displays the mean Area Under the Curve (AUC) of the signal-rejection curve calculated from 10 million test points; standard deviations are in parentheses. The mean expected significance of a discovery (in units of Gaussian σ) is given for 100 signal events and 5000 background events with a 5% relative uncertainty.

	AUC		
Technique	Low-level	High-level	Complete
NN	0.789 (0.0010)	0.792 (0.0002)	0.797 (0.0004)
NN ensemble	0.791	0.793	0.798
DNN	0.798 (0.0001)	0.798 (0.0001)	0.802 (0.0001)
DNN ensemble	0.798	0.798	0.803
	Discovery significance		
Technique	Low-level	High-level	Complete
NN	2.57σ (0.006)	2.92σ (0.006)	3.02σ (0.008)
NN ensemble	2.61σ	2.96σ	3.06σ
DNN	3.16σ (0.003)	3.24σ (0.003)	3.37σ (0.003)
DNN ensemble	3.18σ	3.26σ	3.39σ

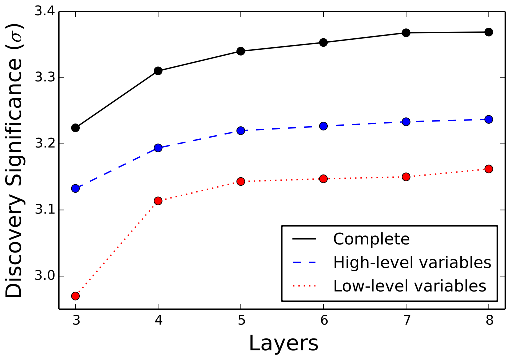

FIG. 4. Effect of network depth on discovery significance. These networks were trained with the hyperparameters optimized for the deep network.

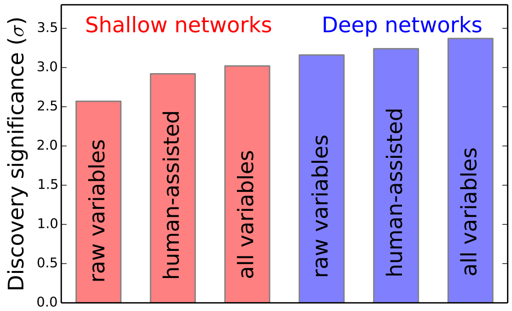

FIG. 5. Comparison of discovery significance for the traditional learning method (left) and the deep learning method (right) using the low-level variables, the high-level variables and the complete set of variables.

DISCUSSION

As expected, the high-level variables derived by physicists clearly capture features of the data that are useful for classification. But while the shallow neural networks perform poorly on the low-level variables alone, deep networks trained on the low-level variables perform nearly as well as deep networks trained on the complete set of variables (even though the Bayesian hyperparameter optimization was performed on networks trained with the complete set). The deep networks are able to learn most of the discriminative information contained in the high-level variables from the low-level variables alone. Note that in the case of the high-level m_{MMC} variable, information regarding the known mass of the τ lepton is included in the calculation; this information is not available from the low-level variables, which may explain in part why the DNN with only low-level variables does not completely match the performance of the DNN with the complete set.

It is also interesting to note that the deep networks learn *additional* information from the low-level variables that is not contained in the high level variables, as evidenced by the deep networks performing better than the shallow networks on both the low-level variables and the complete variables; in fact the deep network with the low-level variables alone performs better than the shallow network trained on the complete set.

A competition hosted through Kaggle recently challenged participants to build classifiers for a similar machine learning task, where one τ lepton decays hadronically to a jet of mostly visible particles. However, the Kaggle data set contains slightly different features and only 250 thousand training examples, compared to the 40 million training examples used here. To avoid overfitting with such a small training set, one would have to use very small or heavily-regularized classifiers. Since an arbitrary quantity of training data can be generated through simulations, this is an unnecessary handicap.

CONCLUSION

We have demonstrated that deep neural networks lead to significantly better performance on detecting $H \to \tau^+\tau^-$ decay events from background compared to shallow networks containing the same number of tunable parameters and optimized in the same manner. Furthermore, the Bayesian optimization algorithm decided for itself that the best neural network depth was *eight layers*, the maximum that we had set prior to starting the algorithm. The deep networks trained on the low-level variables performed *better* than shallow networks trained on the high-level variables engineered by physicists, and almost as well as the deep networks trained high-level variables, suggesting that they are *automatically* learning the discriminatory information contained in the physicist-derived high-level variables. The improvement in discovery significance is equivalent to an increase in the accumulated dataset of 25%.

METHODS

Neural network classifiers were trained with rectified linear hidden units, a logistic output unit, and cross-entropy loss. Network parameters were trained using stochastic gradient descent with mini-batches of 100 examples. A momentum term increased linearly from an initial value to some final value over a specified number of epochs. The entire dataset of 80 million samples was normalized prior to any training; for those features with a skewness greater than 1.0, a small value of 10^{-8} was added, the logarithm was taken, then the values were normalized.

Computations were performed using machines with 16 Intel Xeon cores, 64 GB memory, and NVIDIA Titan or Tesla C2070 graphics processors. All neural networks were trained using the Pylearn2 and Theano software packages [14, 15]. Bayesian optimization was performed with the Spearmint software package using a Gaussian Process model, running 20 experiments in parallel, with 20 Markov Chain Monte Carlo iterations and a burn-in of 50. The test cross-entropy error was used as the objective function for the Bayesian optimization.

The space of possible hyperparameters for the deep networks had six dimensions: number of layers (2 to 8), number of hidden units per layer (100 to 500), learning rate decay factor ($1+10^{-9}$ to $1+10^{-3}$, log scaled), initial momentum (0 to 0.5), final momentum (0.001 to 0.5, log scaled), epochs until momentum saturation (20 to 100). The initial learning rate was 0.01. The space of possible hyperparameters for the shallow networks also had six dimensions: number of hidden units per layer (100 to 56234, log scaled), initial learning rate (10^{-7} to 10^{-2}, log scaled), learning rate decay factor ($1 + 10^{-9}$ to $1 + 10^{-3}$, log scaled), initial momentum (0 to 0.5), final momentum (0.001 to 0.5, log scaled), epochs until momentum saturation (20 to 100).

The best single deep network used eight layers, with 274 hidden units in each of the seven hidden layers. The momentum parameter began at 0 and increased linearly before saturating at 0.996 by epoch 13. The learning rate decayed by a factor of 1.00000051371 after each minibatch update. Experiments with additional hidden layers did not improve performance, but the parameter space for deeper networks was not explored thoroughly. The best shallow network had a single hidden layer of 693 rectified linear units, with an optimized initial learning rate of 0.006, a learning rate decay factor of 1.00000000894, and the momentum remaining constant at 0.5 throughout training. The initial weights of both the shallow and deep networks were drawn from a normal distribution with zero mean and a standard deviation of 0.1 for the first layer, 0.001 for the last layer, and 1.0 divided by the square root of the number of inputs for the other layers.

Experiments using the dropout algorithm and other stochastic methods like adding Gaussian noise to the neuron activations [16, 17] did not improve performance. These approaches help with regularization, but they generally make it more difficult for the network to fit to the training data. Our training data set is large, so the primary challenge is learning rather than avoiding overfitting.

ACKNOWLEDGMENTS

We are grateful to David Rousseau and Kyle Cranmer for their insightful comments, Michael Gelbart for his help with the Bayesian optimization, and Yuzo Kanomata for computing support. We also wish to acknowledge a hardware grant from NVIDIA, NSF grant IIS-0513376, and a Google Faculty Research award to PB.

REFERENCES

[1] G. Aad, T. Abajyan, B. Abbott, J. Abdallah, S. A. Khalek, A. A. Abdelalim, O. Abdinov, R. Aben, B. Abi, M. Abolins, O. S. AbouZeid, H. Abramowicz, H. Abreu, B. S. Acharya, L. Adamczyk, and others, Science **338**, 1576 (2012).

[2] D. Abbaneo, G. Abbiendi, M. Abbrescia, S. Abdullin, A. Abdulsalam, B. S. Acharya, D. Acosta, J. G. Acosta, A. Adair, W. Adam, N. Adam, D. Adamczyk, T. Adams, M. R. Adams, A. Adiguzel, and others, Science **338**, 1569 (2012).

[3] G. Montfar, R. Pascanu, K. Cho, and Y. Bengio, arXiv:1402.1869 [cs, stat] (2014), arXiv: 1402.1869.

[4] S. Hochreiter, *Recurrent Neural Net Learning and Vanishing Gradient* (1998).

[5] Y. Bengio, P. Simard, and P. Frasconi, Neural Networks, IEEE Transactions on **5**, 157 (1994).

[6] P. Baldi, P. Sadowski, and D. Whiteson, Nature Communications **5** (2014), 10.1038/ncomms5308.

[7] J. Alwall *et al.*, JHEP **1106**, 128 (2011), arXiv:[arXiv:1106.0522] [hep-ph].

[8] T. Sjostrand *et al.*, JHEP **05**, 026 (2006), [arXiv:hep-ph/0603175].

[9] S. Ovyn, X. Rouby, and V. Lemaitre, (2009), arXiv:0903.2225 [hep-ph].

[10] A. Elagin, P. Murat, A. Pranko, and A. Safonov, Nucl.Instrum.Meth. **A654**, 481 (2011), arXiv:1012.4686 [hep-ex].

[11] J. Snoek, H. Larochelle, and R. P. Adams, in *Advances in Neural Information Processing Systems 25*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc., 2012) pp. 2951–2959.

[12] M. A. Gelbart, J. Snoek, and R. P. Adams, in *Uncertainty in Artificial Intelligence, Proceedings of the Thirteenth Conference* (Quebec City, Canada, 2014).

[13] G. Cowan, K. Cranmer, E. Gross, and O. Vitells, Eur.Phys.J. **C71**, 1554 (2011), arXiv:1007.1727 [physics.data-an].

[14] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, in *Proceedings of the Python for Scientific Computing Conference (SciPy)* (Austin, TX, 2010) oral Presentation.

[15] I. J. Goodfellow, D. Warde-Farley, P. Lamblin, V. Dumoulin, M. Mirza, R. Pascanu, J. Bergstra, F. Bastien, and Y. Bengio, *Pylearn2: a machine learning research library*, arXiv e-print 1308.4214 (2013).

[16] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, arXiv:1207.0580 (2012).

[17] P. Baldi and P. Sadowski, Artificial Intelligence **210**, 78 (2014).