CMS PAS SUS-11-018

Contact: cms-pag-conveners-susy@cern.ch 2012/03/10

Search for supersymmetry in events with opposite-sign dileptons and missing energy using an ANN

The CMS Collaboration

Abstract

In this note a search for supersymmetry (SUSY) is presented in events with two opposite sign isolated leptons in the final state, accompanied by hadronic jets and missing transverse energy. Advanced multivariate techniques, and in particular Artifical Neural Networks, are deployed in order to discriminate between possible SUSY signals from the Standard Model backgrounds. The analysis uses a data sample collected with the CMS detector during the 2011 LHC run and corresponding to an integrated luminosity of 2.2 fb^{-1}

1 Introduction

With the very successful 2011 LHC run over 2.2 fb^{-1} in pp collisions have been collected with the CMS experiment. This dataset is used to search for the presence of Supersymmetry (SUSY) in events with two opposite-sign leptons (electrons and/or muons) in the final state utilizing an Artificial Neural Network. The 2.2 fb^{-1} data sample, out of the 5 fb^{-1} collected in total, exhibits lower event pile-up and hence has potentially lower, and different, systematic uncertainties than then remaining 2.8 fb^{-1} data set. There are many possible ways to produce two leptons in Supersymmetric events. For example, squarks or gluinos can be pair produced in pp collisions, leading either to dileptons in the cascade decay chain of one of the supersymmetric particle (sparticle) or one or more leptons in the cascade decay chains of both sparticles. We assume that an additional R-parity symmetry is conserved, leading to a stable lightest sparticle (LSP) and hence to a significant missing energy signature. Depending on the mass splitting between the gluinos or squarks and the LSP, the missing energy signature can be large or small. Up to now, the criteria used in CMS to discriminate SUSY signals from the very large Standard Model (SM) backgrounds has tended to require several high transverse momentum jets and large missing energy. Compared with previous CMS searches [1, 2], relaxed criteria on missing transverse energy (MET) is achieved which enables this analysis to probe additional regions of phase-space not previously investigated.

This summary is organized as follows : In section 2 the event samples, triggers, and event selection criteria are presented. In section 3 the main idea and properties of Artificial Neural Networks (ANN) is discussed. In addition, the ANN construction, optimization and expected performance on simulated events is presented. In Section 4, the method used to obtain the ANN prediction for SM processes is shown and in Section 5 the systematic uncertainties along with their estimation is discussed. In Section 6, detailed comparisons of the ANN input variables and the ANN output between data and simulated events are presented in both signal-enriched and background-enriched regions. Finally the results in the context of the Minimal Supersymmetric extension of the Standard Model (MSSM), and the Constrained MSSM (CMSSM) [6] are shown.

2 Event Samples, Trigger and Event Selection

The CMS experiment collects data using a two-level trigger system, the first-level hardware trigger (L1) [7] and a high-level software trigger (HLT) [8]. Data events are selected using a set of dilepton triggers which require the presence of at least two leptons, either two electrons or two muons or an electron-muon pair.

Muon candidates are reconstructed [9] by combining the information from the inner tracking system, the calorimeters, and the muon system. Electron candidates are reconstructed [10] by combining the information from the Electromagnetic Calorimeter (ECAL) with the silicon tracker, using shower shape and track-ECAL cluster matching variables in order to increase the sample purity. Jets are reconstructed using the anti-k_{T} clustering algorithm [11] with a size parameter $R = 0.5$. Calorimeter (CALO) jets are reconstructed by clustering four-momentum vectors, formed from the energy deposits in calorimeter towers. A calorimeter tower consists of one or more Hadronic Calorimeter (HCAL) cells and the geometrically corresponding ECAL crystals. Jets are corrected using factors derived from simulation; to correct for any differences in the energy response between simulation and data, a residual correction factor derived from data is applied to jets in data [12].

In general, missing transverse energy (MET), is the negative vector sum of the transverse momenta of all final-state particles reconstructed in the CMS detector. The calorimetric missing transverse energy is calculated using the energies contained in calorimeter towers along with their directions. The MET is corrected for the presence of muons. The total transverse energy of the event is calculated as the scalar sum of the transverse energies over all calorimeter towers and muons. The total hadronic transverse energy, (HT), is computed as the scalar sum of the transverse energies of all reconstructed jets in the event satisfying the jet selection criteria.

Simulated pp collision events are produced with the PYTHIA 6.4.22 [13] (using underlying event tune Z2) and MADGRAPH 4.4.24 [14] generators , and processed with a simulation of the CMS detector response based on GEANT4 [15]. Simulated events are reconstructed and analyzed in the same way as data events. Simulated event samples are also used to train the ANN and for extrapolating data-driven background estimates from a background-enriched control region to the expected signal-enriched region. Finally, the simulation is used to perform extensive quality control checks of the data samples collected the "signal" and "background" enriched regions, and for estimating systematic uncertainties.

Non-collision backgrounds are removed by applying quality cuts ensuring the presence of a well-reconstructed primary vertex [16]. Events are required to have at least two leptons with $p_T > 20$ GeV and $|\eta| < 2.4$, and at least two jets with $p_T > 30$ GeV and $|\eta| < 2.4$. All data results, and data vs MC comparisons, have a lepton p_T cut at 20 GeV in order to be above the trigger efficiency plateau ($>$ 99% efficiency). Jets are required to satisfy the quality criteria described in [17]. Leptons are required to be isolated from significant energy deposits and tracks in a cone of radius $\Delta R = 0.3$ around the direction of the lepton. The relative combined isolation, defined as $I^{comb.}_{rel.} = (\sum_{tracks} P_T + \sum_{ECAL} E_T + \sum_{HCAL} E_T)/P_T$, is required to be < 0.15 for muons and < 0.08 for electrons.

3 Methodology for event selection

3.1 Artificial Neural Networks

When one wishes to discriminate "signal" in a sample that consists of both "signal" and "background", one common technique is the application of several sequential selection cuts in variables that characterize those events. In the simple case of discriminating and uncorrelated variables describing the two populations, placing a set of "cuts" on these variables to characterize an event as "signal" or "background" is straightforward. However, when the events of interest are characterized by a set of correlated variables, which is a very common case, then the selection optimization using linear cuts can be both non-trivial and inefficient. A powerful, well-known technique alleviating this problem, is that of Artificial Neural Networks (ANN), which can be used to discriminate between the signal and background populations [18, 21]. The main interest for this work is the case of feed-forward back-propagation networks, which are able to induce sophisticated cuts that physicists cannot easily deduce by hand. This work uses a software package known as TMVA [22] to implement the ANN.

3.2 ANN training

Due to the natural presence of isolated leptons, the main SM background contributions to this analysis involve the production of $t\bar{t}$, and Z+jets. QCD multijet processes with two fake leptons and W+jets events with one fake lepton can also be part of the background, but are significantly reduced by the preselection criteria described below. Finally, two leptons in the final state are produced by WW, WZ, ZZ events but their contributions are found to be negligible compared to the above main ones.

The loose preselection criteria are listed in Table 1 and are imposed before the ANN training, so as to reject the vast majority of the background, whilst retaining most of the signal.

Missing transverse energy, MET > 30 GeV
Number of jets, $N_{jets} > 1$
Fraction of lepton to total transverse energy, $\frac{E_T^{lepton}}{\sum E_T} < 0.3$
$\Delta R = \sqrt{\Delta\phi^2 + \Delta\eta^2}$ between lepton and closest jet ,$\Delta R_{1^{st}lepton-jet} > 0.2$ and $\Delta R_{2^{st}lepton-jet} > 0.2$
Invariant mass of two leptons, $M_{ll} > 10$ GeV

Table 1: Preselection criteria, applied prior to ANN training

The preselection cuts on the missing transverse energy, the fraction of the transverse energy carried by the leptons, and the number of jets, significantly reduce the Z+jets, $t\bar{t}$ and W+jets backgrounds. The cuts on the di-lepton invariant mass and the ΔR between the leptons and the closest jet reduce significantly the QCD background. In Table 2, we present the reduction that the preselection variables impose on the SM backgrounds and for three characteristics CMSSM benchmark points: LM0 (($m_0 = 200GeV/c^2, m_{1/2} = 160GeV/c^2, \tan\beta = 10, A0 = -400GeV/c^2$)), LM1 (($m_0 = 60GeV/c^2, m_{1/2} = 250GeV/c^2, \tan\beta = 10, A0 = 0GeV/c^2$)) and LM6 (($m_0 = 85GeV/c^2, m_{1/2} = 400GeV/c^2, \tan\beta = 10, A0 = 0GeV/c^2$)).

Sample	After Event Selection	After Preselection	Efficiency
$t\bar{t}$	7455	1541	0.21
Z+Jets	32336	1737	0.05
W+Jets	1584	50	0.03
WW	532	6.5	0.01
WZ	773	8.8	0.01
ZZ	541	5.9	0.01
QCD	41490	0.003	5×10^{-8}
Total SM Bkg.	84711	3349	0.04
LM0	2703	1263	0.47
LM1	356	190	0.55
LM6	33	18	0.56

Table 2: Expected number of events for 2.2 fb^{-1} for signal and background after the preselection criteria are applied. For the CMSSM benchmark points the NLO cross sections have been used.

The ANN training samples are based on simulated events; the network architecture and an "early stopping" strategy are used to avoid overtraining as described in Appendix A. A mixture of $t\bar{t}$, Z+jets, W+jets, and QCD simulated samples, weighted according to the expected number of events passing the preselection cuts, are used as the SM background samples to train the ANN. The CMSSM benchmark point LM0 ($m_0 = 200GeV/c^2, m_{1/2} = 160GeV/c^2, \tan\beta = 10, A0 = -400GeV/c^2$) is used as the signal sample to train the ANN, since it has kinematical properties typical of several low MET SUSY models. While this low mass point has already been excluded by previous CMS searches [1, 2], the training only utilizes the generic kinematic features of the sample. The ANN performance is tested on a variety of different new physics models and reasonable sensitivity is obtained, as shown in Appendix A.

Several topological and kinematical variables were considered according to their potential to discriminate SM backgrounds versus possible SUSY signals, including the correlations between the variables. The kinematical variables studied are generic and based on the general properties of Supersymmetric processes in many SUSY models. The decays of the produced sparticles result in final states with two neutralino LSPs, which escape the detector thus creating large missing transverse energy. Also the cascade decays of SUSY particles typically yield leptons of lower momenta that carry a relatively small fraction of the total event energy, compared to leptons produced in SM processes. On the other hand, due to the possible long cascade decay chains, the number of jets in SUSY events can be much higher compared to those from SM events. The complete list of the variables considered, along with a brief explanation for each one is shown in Appendix A.

Using different sets of input variables, taking into account their discriminating power, their correlations, and possible systematic uncertainties, several ANNs were constructed and compared and an optimum ANN chosen. The differences in performance were studied and quantified in terms of the efficiency as a function of background rejection and as a figure of merit (FOM), defined as $\frac{S}{\sqrt{B}}$. A network having seven input variables, those which had the smallest degree of correlation amongst themselves and which also possessed the highest discriminating power, showed the best performance. Table 3 lists the seven input ANN variables along with their description and their relative importance to the ANN after training.

Variable	Description	ANN Importance (%)
MET	Missing transverse energy	17
N_{jets}	Number of jets	19
$\frac{E_T^{lepton}}{\sum E_T}$	Ratio of energy of the dilepton system to the total transverse energy	1
Jet p_T	Primary and secondary jet p_Ts	13 - 23
M_{ll}	Dilepton mass	12
Invariant Mass M_T	Invariant event mass	15

Table 3: Seven event, lepton and jet related variables used for the ANN construction.

The relative ANN importance is defined as the sum of the weights-squared of the connections between the variable's neuron in the input layer and the first hidden layer. The chosen seven input variables are also shown in Figures 1-2 for simulated and data events, and after the preselection cuts are applied. Data and simulation are consistent and agree with each other given the statistical and systematic uncertainties. The final ANN output function after training is given in Appendix C.

4 ANN Output for SM Background

The most important step of the analysis is to robustly estimate the ANN output distribution for the SM background expectation, along with its systematic uncertainty. Once this is accomplished, one can quantify the level of agreement (or the significance of a possible excess) between the ANN output from data and the SM prediction in the expected signal region. The approach used to estimate the ANN prediction for the SM-only hypothesis from data is the following:

- A control region (CR) is defined by inverting two of the preselection cuts, the total missing transverse energy and the fraction of transverse energy carried by the dilepton system. This region is chosen so that it is dominated by SM processes, and indeed the contamination from the CMSSM benchmark point LM0 is estimated to be $< 2\%$, and $< 0.02\%$ for LM6. The ANN output distribution in the control region is then determined using data, $ANN(SM)_{CR}^{Data}$.
- An extrapolation ratio, $R_{Ext.} = \frac{ANN(SM)_{SR}^{MC}}{ANN(SM)_{CR}^{MC}}$ obtained from simulated events, is defined for each bin in the ANN output distribution as the ANN output for the SM-only hypothesis in the signal region (SR) divided by the ANN output for the SM-only

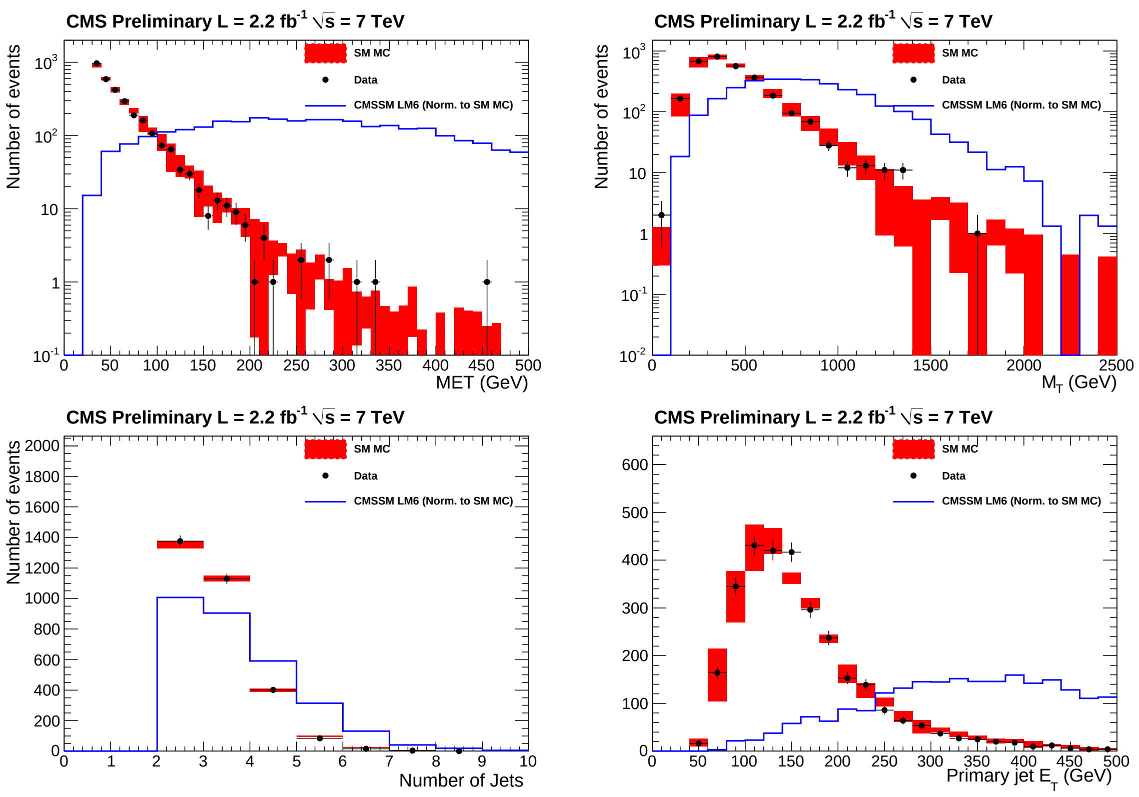

Figure 1: The first four out of the seven input ANN variables for simulated SM backgrounds (red), CMSSM LM6 benchmark point (blue), and data (black points) normalized to the same number of events. Statistical and systematic errors on the data and simulated events are shown.

hypothesis in the control region.

- The ANN output from data in the control region, where only SM physics is assumed to be present, is multiplied by the extrapolation factor, $R_{Ext.}$, in order to predict the ANN output SM in the signal region, $ANN(SM)^{Prediction}_{SR}$.

$$ANN(SM)^{Prediction}_{SR} = ANN(SM)^{Data}_{CR} \times \frac{ANN(SM)^{MC}_{SR}}{ANN(SM)^{MC}_{CR}} \tag{1}$$

The extrapolation factor, $R_{Ext.}$, as extracted from simulated events is shown in Figure 3.

The control region is further subdivided in a $t\bar{t}$ enriched one with MET $>$ 30 GeV and $M_{ll} >$ 105 GeV OR $M_{ll} <$ 75 GeV, denoted as “Control Region A”, and a Z+jets enriched one with MET$<$ 30 GeV OR 75 GeV $< M_{ll} <$ 105 GeV, denoted as “Control Region B”. These are not used in the analysis, however they provide additional sanity tests.

5 Systematic Uncertainties

Systematic uncertainties of the ANN output prediction for the SM-only hypothesis, obtained as described in section 4, are estimated with simulated data using the following procedure:

- A systematic effect is introduced into the simulated data for all events in the sample before any preselection is applied.
- The nominal SM extrapolation factor $R_{Ext.}$ is used to obtain a new ANN output

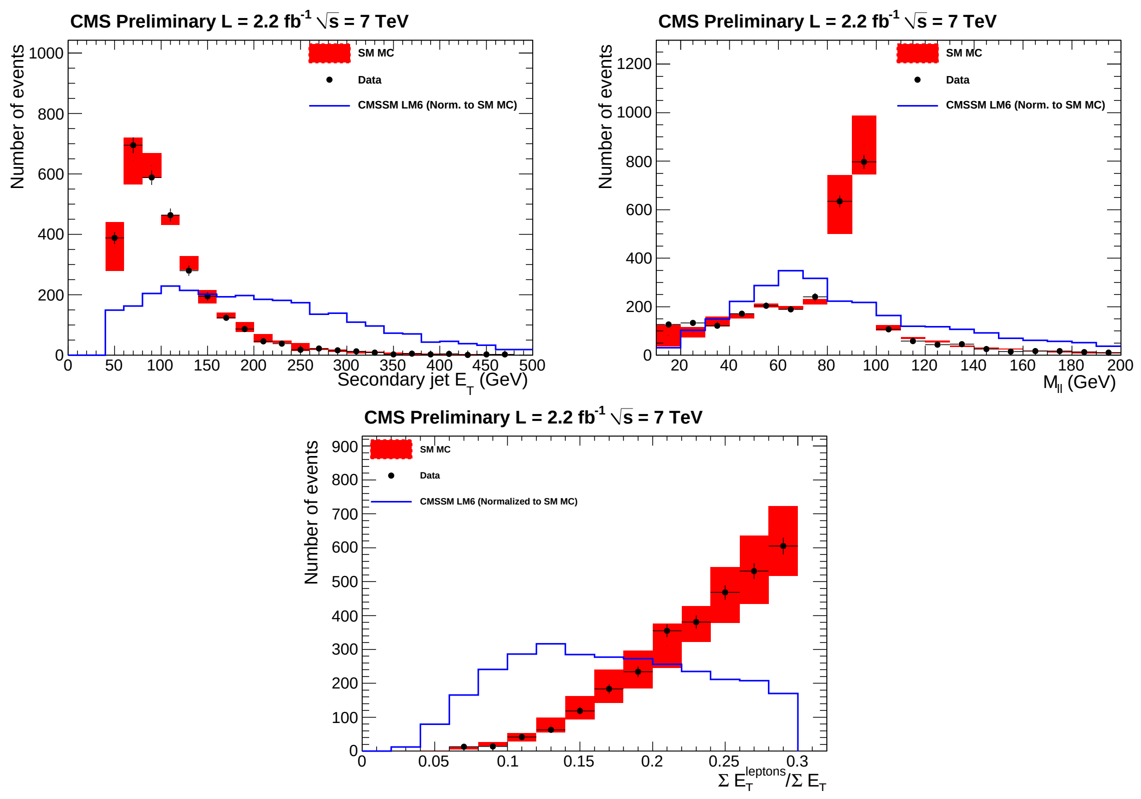

Figure 2: The remaining three out of the seven input ANN variables for simulated SM backgrounds (red), CMSSM LM6 benchmark point (blue), and data (black points) normalized to the same number of events. Statistical and systematic errors on the data and simulated events are shown.

prediction for the signal region corresponding to the systematic effect under study.

- The ANN output prediction, corresponding to the systematic alteration, is compared against the ANN output for the original sample, without any systematic effects introduced. The relative difference in ANN outputs for each bin, is assigned as a bin-by-bin systematic uncertainty.
- For each bin, the relative differences for all systematic effects studied are added in quadrature. This results in a total systematic uncertainty for each bin in the ANN output prediction.

The overall systematic uncertainties, corresponding to the seven input variables used for the ANN construction as well as the cross sections of the SM backgrounds, are shown in Table 4.

The magnitude of the systematic alterations for the jet and MET (and M_T) scales are taken from dedicated CMS measurements [24, 25]. The migration of events from the one-jet to the two-jet bin (0.5%) is an estimate of the difference (0.5%) between data and simulated events. The ratio of the lepton to the total transverse energy (4%) is a conservative estimate of the difference ($\sim$ 3%) between data and simulated events in the control region. The dilepton mass scale uncertainty (1%) is taken from the CMS measurements of the Z peak [29].

Finally, the relative fraction of $t\bar{t}$ and Z+jets backgrounds are observed to vary both as a function of the ANN cut, as well as across the signal and control regions. However, the shape differences between the two regions are small. In order to account for any remaining differences , not already accounted for in the extrapolation factor $R_{Ext.}$, the cross sections of all background

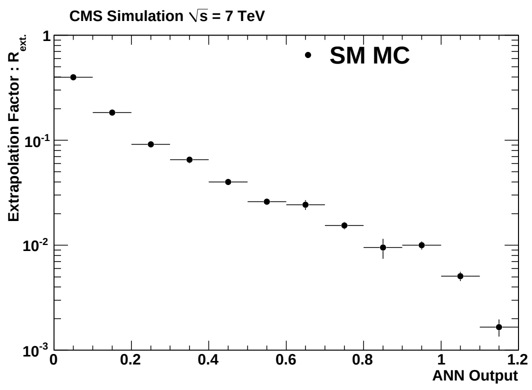

Figure 3: Extrapolation factor $R_{Ext.}$, as obtained from simulated events.

Systematic	Magnitude	% of Total for ANN > 0.8
Missing Transverse Energy MET	±10%	17
Number of Jets N_{jets}	±0.5%	<1
$\frac{E_T^{lepton}}{\sum E_T}$	±4%	10
Primary and Secondary jet p_T	±5%	11
$t\bar{t}$ cross section	±10%	4
Dilepton Mass M_{ll}	±1%	3
Invariant Mass M_T	±10%	9
QCD cross section	±20%	2

Table 4: Systematic uncertainties considered in the predicted background, along with their magnitude, and the impact they have on the final ANN output prediction when the signal selection cut at 0.8 is applied.

components are left to vary within their uncertainties, taken from the recent CMS measurements for the $t\bar{t}$ [27], and QCD [26] events. The Z+jet cross section uncertainty ($< 3\%$) [28], and the W+jet cross section uncertainty ($< 3\%$) [28] produce a negligible effect and hence are neglected.

6 Performance of the ANN

The ANN output after the training is shown in Figure 4 for the signal (blue) and SM background (red) populations; the efficiency and purity of the selected samples is also shown as a function of the ANN output cut.

When statistical and systematic uncertainties are taken into account, the ANN output cut that yields the maximum FOM is $P > 0.8$. The expected number of events after imposing the ANN output cut are presented in Table 5.

As noted previously, the ANN has been trained with a new physics (NP) hypothesis that is definitively not what nature has chosen. Hence, it is of great interest to examine the power that this ANN has in discriminating signal hypotheses for which it has not been trained. Examples of the ANN performance for different NP hypotheses are shown and discussed in Appendix A. Those results demonstrate that the ANN is able to discriminate a variety of different SUSY models from the SM-only hypothesis. Since this is a preliminary study, the choice of a single signal training sample is sufficient to demonstrate the proof-of-principle of this method.

For illustration purposes, Figure 5 shows a comparison between data and simulated events

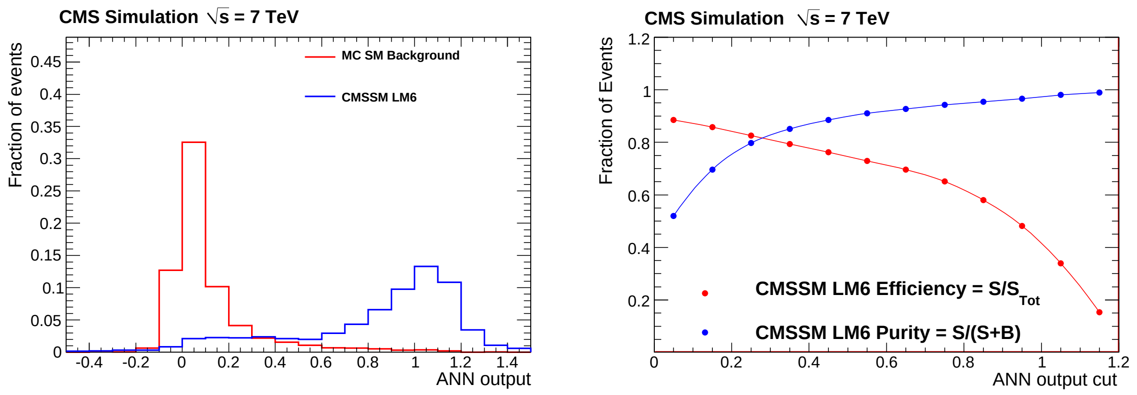

Figure 4: Left : ANN output for the SM background (red line) and CMSSM LM6 benchmark point (blue line) Right : Efficiency (red) and purity (blue) vs ANN output cut for CMSSM LM6 benchmark point, after the event and preselection criteria are applied.

Sample	ANN > 0.8
$t\bar{t}$	50
Z+Jets	39
W+Jets	1.5
WW	0.27
WZ	0.08
ZZ	0.10
QCD	0.001
Total SM Bkg.	91
LM0	367
LM1	115
LM6	12.2

Table 5: Expected number of events for 2.2 fb^{-1} for signal and SM backgrounds for the ANN probability cut at $P > 0.8$ which yields maximum Figure of Merit. The NLO cross sections are used for LM0, LM1 and LM6.

of the ANN output distributions in the control regions. Good agreement between data and simulation is observed.

Good agreement between data and simulation for the ANN input variables is demonstrated in Appendix B. Comparisons between data and simulation for the seven different ANN input variables in the control region, help validate that the simulation is sufficiently good with which to train the ANN, and adequate to be used for the estimation of systematic uncertainties. It is important to note that, within the signal region, this analysis does not actually use the SM-only ANN output from simulation to search for a possible signal in the data. Rather as previously described, this work uses the ANN output distribution obtained from data in the control region, and extrapolates that distribution, using the bin-by-bin factors $R_{Ext.}$ from the simulation, into the signal region to obtain an ANN output prediction for the SM-only hypothesis.

7 Results

In this section the results of the analysis on the 2.2 fb^{-1} of data collected during the 2011 LHC Run with the CMS detector are presented. First, the ANN output prediction for the SM-only hypothesis, with statistical and systematic uncertainties taken into account, is presented along with its comparison with observations. Then, in the absence of any significant excess of events

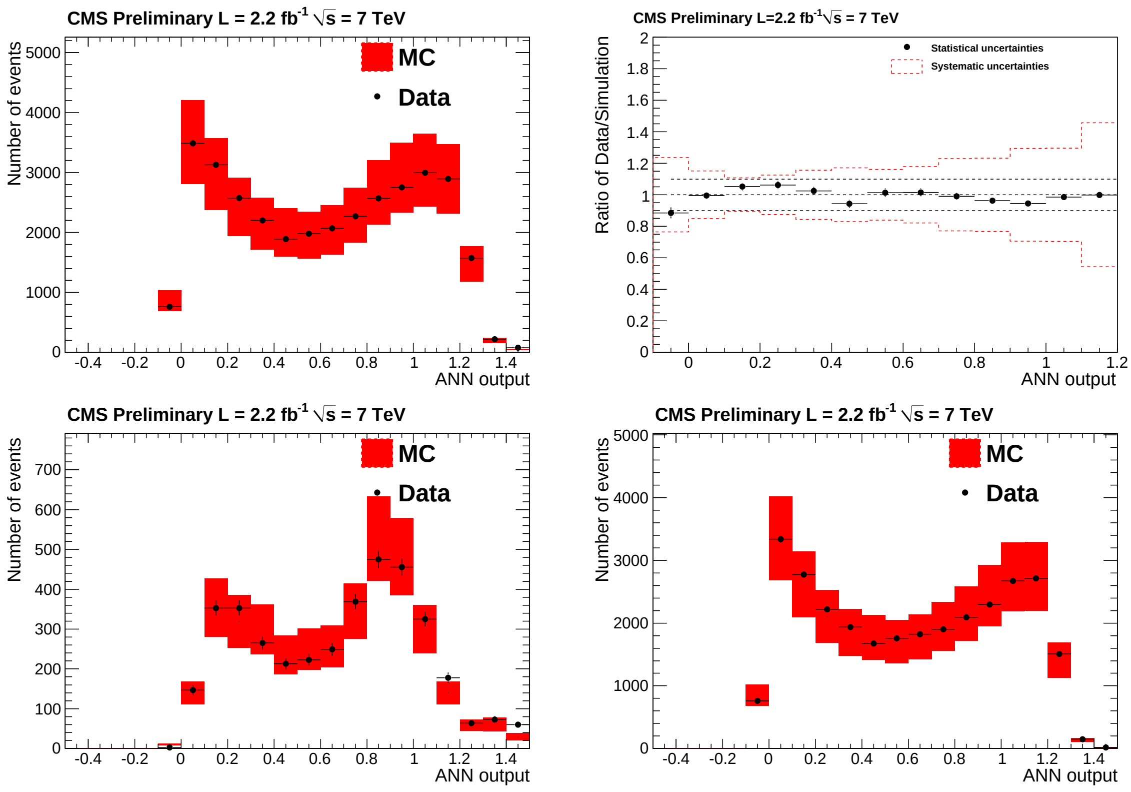

Figure 5: Data (black points) vs MC (red lines) comparisons of the ANN output distributions in the various control regions. Top : The ANN output in the control region used in order to perform the extrapolation with 20% systematic uncertainties included (left), and the ratio between data and simulated events (right) with both statistical (black error bars) and systematic (red bands) uncertainties shown. Bottom : Two additional control regions not used in the analysis, but used for sanity checks : $t\bar{t}$ enriched (left), Z+jet enriched (right) with 20% systematic uncertainties included. The 20% systematic uncertainty is the smallest bin-by-bin systematic error used for illustration purposes.

observed in data with respect to the SM expectation, exclusion limits are set for a simplified model involving gluino pair production.

7.1 Standard Model ANN output Prediction

Figure 6 shows the comparison between the SM ANN prediction (obtained as described in Sections 4 and 5), with statistical and systematic uncertainties computed, with the data in the signal region. The agreement between expectation and observation is good and no significant hint of new physics is present.

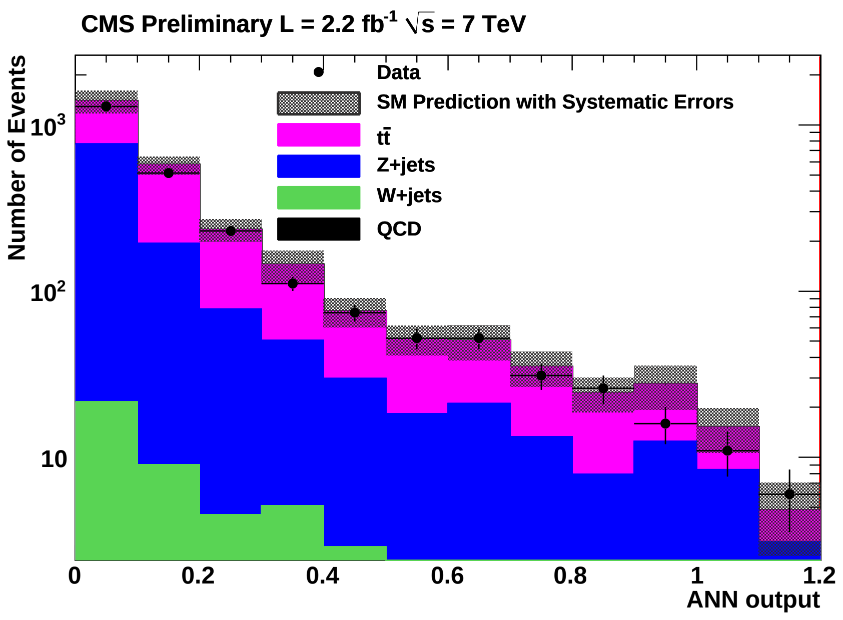

Figure 6: ANN output for the data (black points) and the SM data-driven prediction (red lines) in the signal region.

Table 6 presents the number of observed and predicted events above a certain ANN cut. The error on the expectation includes both statistical and systematic uncertainties. The systematic uncertainties for the cumulative distribution are computed as follows : for each systematic effect pseudo-experiments are generated, and the cumulative number of events above a certain ANN cut is calculated and compared with the expectation under the hypothesis that no systematic effect is present. Then, considering the systematics as uncorrelated, the individual integral differences are added in quadrature in order to compute the total systematic error.

With the optimal ANN cut at 0.8, agreement between the observed number of events and the expected number of events is seen to be compatible at the 68% confidence level.

7.2 Interpretation of the results

The observed and expected number of events are translated into limits on a Simplified Model (SMS) [30],[31] (m_{GL} vs m_{LSP}) plane. The dilepton model used is constructed as follows : one gluino undergoes a direct decay, while the other decays to an intermediate heavy neutralino which decays to a pair of leptons. The leptons are chosen with equal probability as either e,m, or tau. This topology is characterized by the opposite sign lepton edge.

The signal selection efficiency systematic uncertainty is taken into account. The systematics associated with the signal selection efficiency, along with their magnitude, are summarized in

ANN output cut	Data	Expectation
0.00	2413	2605.1± 344.0
0.10	1123	1209.0± 150.2
0.20	611	655.4± 94.3
0.30	381	417.4± 69.7
0.40	270	287.1± 51.4
0.50	196	209.5± 38.7
0.60	144	157.1± 31.9
0.70	92	110.8± 25.6
0.80	61	75.6± 18.8
0.90	35	51.1± 15.9
1.00	19	22.6± 8.4

Table 6: Number of events for the SM Template expectation and data in the signal region, as a function of the ANN output cut for the entire 2.2 fb^{-1} dataset.

Table 7.

Systematic	Magnitude
Lepton Triggers ($p_T > 20$ GeV)	3%
Lepton Isolation	5%
Luminosity	4.5%
ANN selection	4-15%
Total	8-17%

Table 7: Signal selection efficiency systematic uncertainties

The uncertainty on the lepton triggers and the lepton isolation are the same as the ones estimated in the same-sign di-lepton analysis [3]. The uncertainty of the ANN selection results from the systematics considered in Section 5 and for CMSSM benchmark point LM6. The relative ANN selection uncertainty for the signal is lower than the corresponding uncertainty for the background, due mainly to the different ANN shapes for these two populations (signal and background). Results are obtained and presented for the ANN $>$ 0.8 inclusive selection.

Selection	Expected	Observed	95% C.L. Upper Limit
ANN $>$ 0.8	75.6 $\pm$ 18.8	61	17

Table 8: Number of predicted and observed events for 2.2 fb^{-1} for the ANN analysis

The 95% C.L. expected exclusion limit for the ANN inclusive analysis in the Simplified Model plane is shown in Fig. 7. For events close to the diagonal, which exhibit low MET and HT characteristics, the ANN analysis yields higher acceptance than the cut-based analysis by factors of $\sim$ ten and better upper cross section limits by factors of $\sim$ two.

The 95% C.L. upper limits are computed using a frequentist CL$_s$ method with profile likelihood test statistics, and truncated Gaussian distribution for the background expectation [34], [35]. The uncertainties in the NLO cross sections from the parton distribution functions, the choice of the factorization and renormalization scale, and α_S are taken into account for each point, and are evaluated according to the PDF4LHC recommendation [36].

The signal contamination on the background prediction is negligible ($<$ 2% for LM0, $<$ 0.02% for LM6), and a uniform acceptance systematic is assumed for each point.

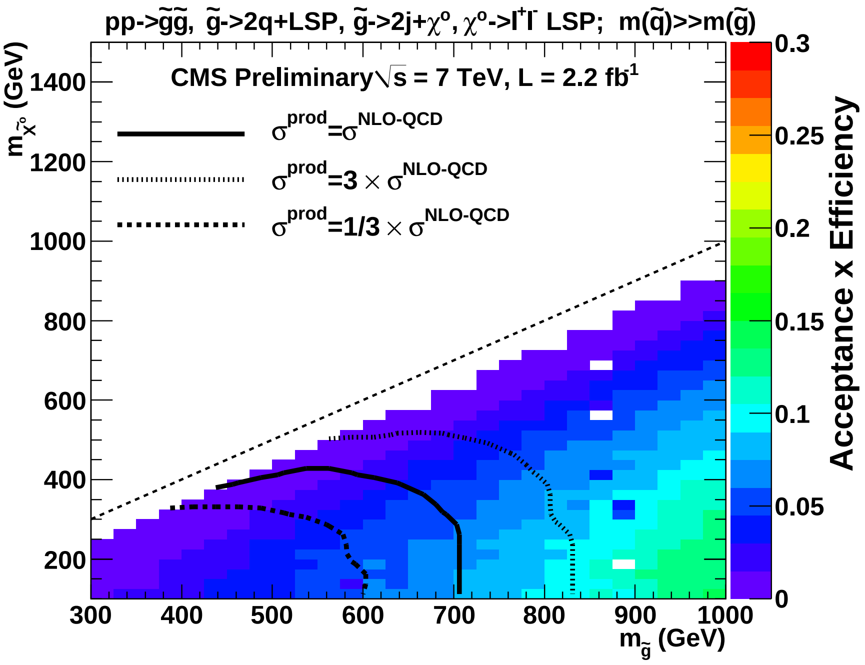

Figure 7: 95% C.L. exclusion limits on Simplified Models of the inclusive ANN analysis (ANN> 0.8).

8 Conclusions

A search for supersymmetry in events with two opposite sign leptons in the final state and with the use of Artificial Neural Networks has been presented, using the 2011 dataset collected with the CMS experiment, corresponding to an integrated luminosity of 2.2 fb^{-1}. This search is independent and complementary to the ones already published [1], [2]. Good agreement is observed between expectation and observation. No significant excess between the SM ANN output prediction in the signal region and the data is observed.

9 Appendix A

ANN candidate input variables : In Table 9 we present a complete list of candidate ANN input variables along with a brief explanation for each one.

Input Variable	Definition
E_T primary lepton	Primary Lepton Transverse Energy
E_T secondary lepton	Secondary Lepton Transverse Energy
M_{ll}	Dilepton Invariant Mass
$\frac{E_T^{lepton}}{\sum E_T}$	Fraction of Event Transverse Energy carried by the Dilepton system
$\Delta\phi_{lepton}$	ϕ difference between the two leading leptons
$\Delta R_{1^{st}lepton-jet}$	ΔR between leading lepton and closest jet
$\Delta R_{2^{nd}lepton-jet}$	ΔR between secondary lepton and closest jet
$M_{1^{st}lepton-jet}$	Invariant Mass between leading lepton and closest jet
$M_{2^{nd}lepton-jet}$	Invariant Mass between secondary lepton and closest jet
Type$_{lepton}$	Dilepton pair type, 1 = $\mu\mu$,2=$e\mu$, 3=ee
N_{jets}	Number of jets
E_T primary jet	Primary Jet Transverse Energy
E_T secondary jet	Secondary Jet Transverse Energy
$\Delta\phi_{jet}$	ϕ difference between the two leading jets
a_T	a_T variable Defined with all event objects (jets and leptons)
M	Invariant Mass of event
M_T	Transverse Invariant Mass of the event
H_T	Hadronic Transverse Energy
MH_T	Missing Hadronic Transverse Energy
MET	Missing Total Transverse Energy
$Min\Delta\phi$	Minimum $\Delta\phi$ between the recoil vector sum of
	all objects (jets and leptons) and the closest object.

Table 9: List of candidate input ANN variables

ANN Training : The network architecture in terms of hidden layers and hidden neurons is shown in Figure 8. In Figure 8 the evolution of the error on the training (red line) and test (blue line) samples is also shown. Training of an ANN is the procedure of finding the global minimum on the "cost" (error) . The error function, E , of an artificial neural network (ANN), is the sum over the networks output neurons and the network training set of events, of the difference between the desired and obtained output:

$$E = \sum_p E(p) = \sum_{jp} (d(p,j) - t(p,j))^2 \tag{2}$$

where p runs over the events of the training set, j is the index of an output neuron, $d(p,j)$ is the desired output of neuron j in event p and $t(p,j)$ is the actual networks output. While training the error on the two statistically independent samples (training and test) is monitored in order to avoid overtraining. Overtraining occurs when the ANN learns the training examples perfectly (error on training sample decreasing) while loses the ability to generalize (error on test sample increasing) . As shown in 8 overtraining occurs at around epoch 200 after which the error on the cost function of the training sample decreases while the one on the test sample starts increasing. The "early-stopping" approach is adopted which stops ANN training when such a trend starts to develop.

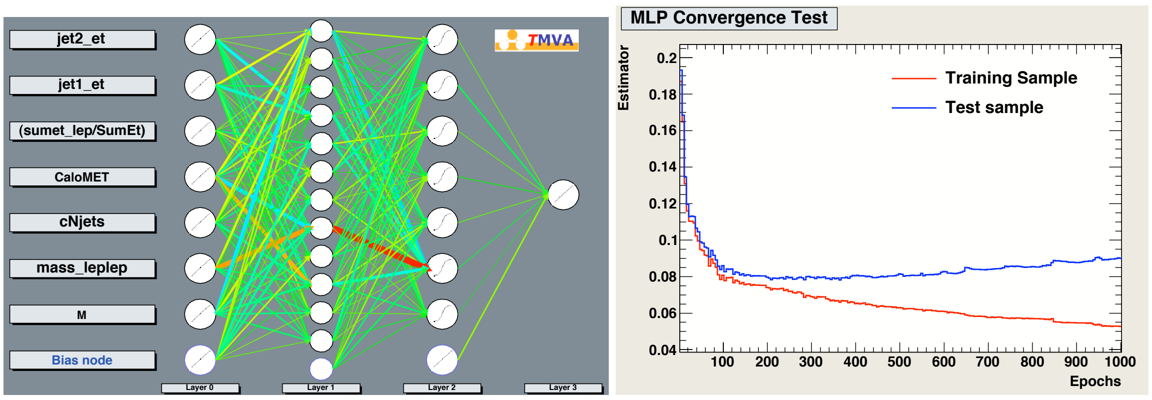

Figure 8: Left : ANN with seven input variables architecture. Right: Error evolution in the cost function minimization procedure for the training (red) and test (blue) populations.

ANN Generalization Ability : The ANN has been trained on a simulated event sample for a specific new physics (NP) model. Since nature might have chosen differently it is of great interest to examine the discriminating power this ANN has on hypothesis it has not been trained with. If the new physics has similar characteristics (high missing transverse energy, large number of energetic jets, large event invariant mass) as the ones we used to train the ANN, then the ANN estimator should be able to discriminate it from the SM background. In general any new physics (NP) should appear as an ”excess” above the ANN output distribution obtained when assuming SM physics only. Depending on the characteristics of the NP the excess will appear closer to unity if NP is more CMSSM-like, or closer to zero if it is more SM-like. The fact that if NP is present a deviation from expectation (in the presence only of SM physics) will appear somewhere in the ANN output distribution can be considered as a great advantage with respect to cut-based analysis, which will not see any hint of NP if its characteristics are SM-like, and hence will not survive the cuts.

For this reason the performance of the ANN estimator is tested on different CMSSM points, LM6 (4) LM1 and LM9 and on Gauge Mediated (GGM) SUSY [32],[33] event samples with different parameter settings. As seen in Figures 9, 10 the shapes of the ANN output distributions for these different models are very similar to the one obtained with the sample used to train the ANN with. Namely, NP is peaking away from zero and close to unity, and in general shows a very different shape than the SM ANN output expectation in the absence of any new physics.

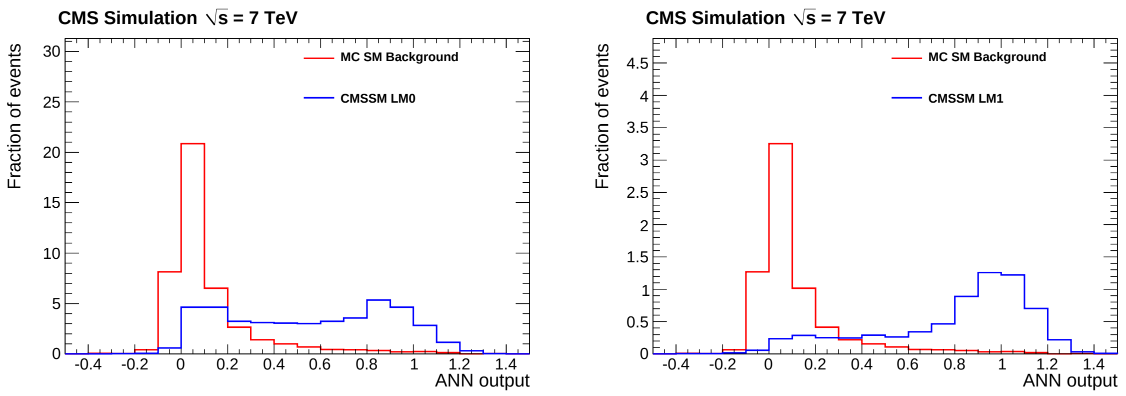

Figure 9: ANN output for the SM background (red) and Top left: CMSSM LM0 benchmark point(blue). Top Right: CMSSM LM1 benchmark point (blue)

Finally, the 95%C.L. expected and observed exclusion limit for the inclusive ANN analysis in the CMSSM plane is shown in Fig. 11. The contamination of the signal in the control region is

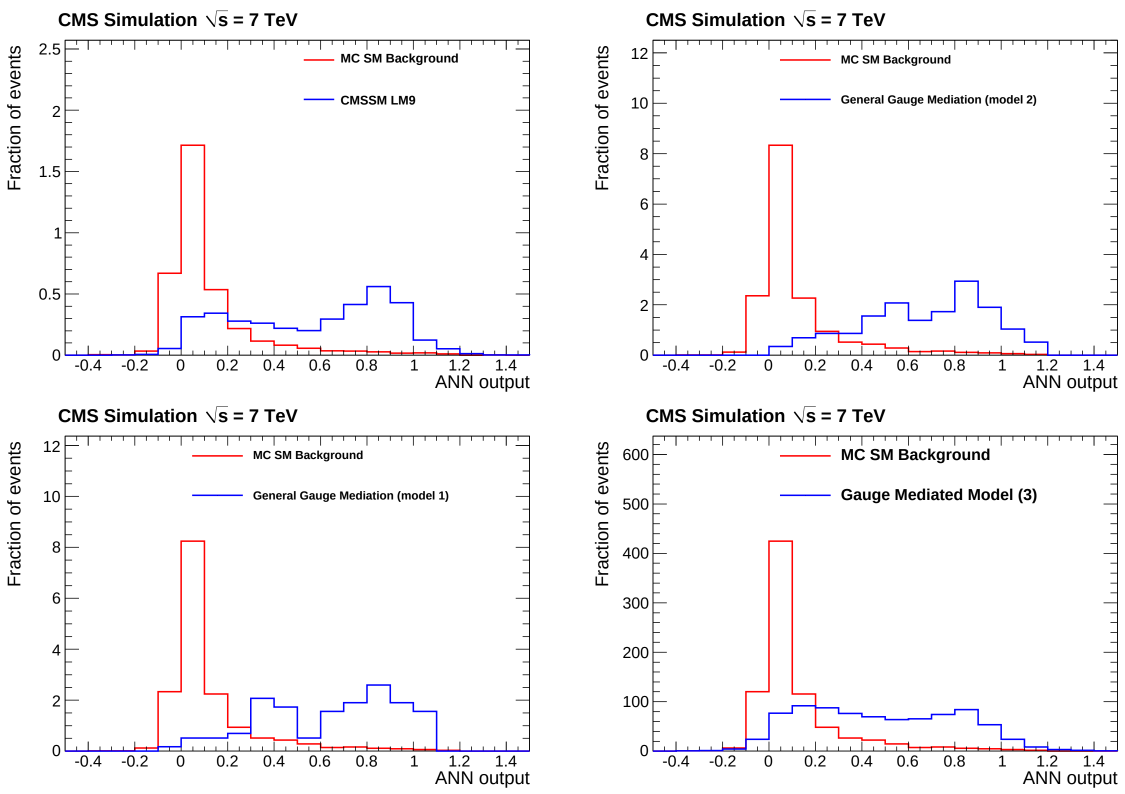

Figure 10: ANN output for the SM background (red) and Top left : CMSSM LM9 benchmark point (blue) Top Right: GGM tanbeta=10 sample (blue) Bottom Left : GGM tanbeta=20 sample (blue) Bottom Right : GMSM sample (blue)

negligible ($< 2\%$ for LM0, $< 0.02\%$ for LM6), and a uniform acceptance systematic is assumed for each point.

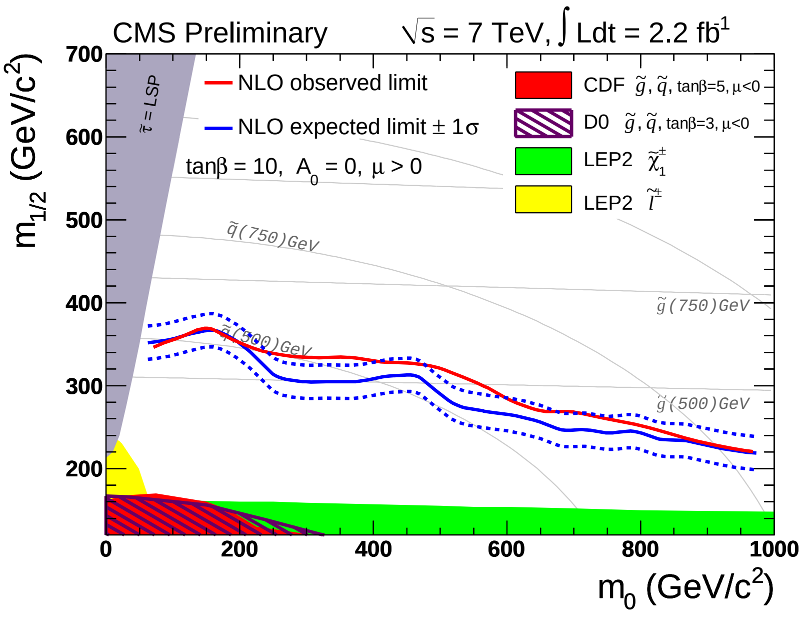

Figure 11: Expected (blue) and observed (red) 95% C.L. exclusion limit for the ANN inclusive analysis (ANN> 0.8)

10 Appendix B

Data vs Simulated event comparisons of ANN input variables in the control region : In Figures 12-13 the comparisons of the ANN input variables between data and simulated events are shown for the CR. Statistical and systematics uncertainties are included. The simulated events are normalized to the total number of the data events. The agreement between data and simulated events of all seven input variables is good, no pathologies or problematic behavior is observed. The di-lepton pair flavours are also in agreement with expectations.

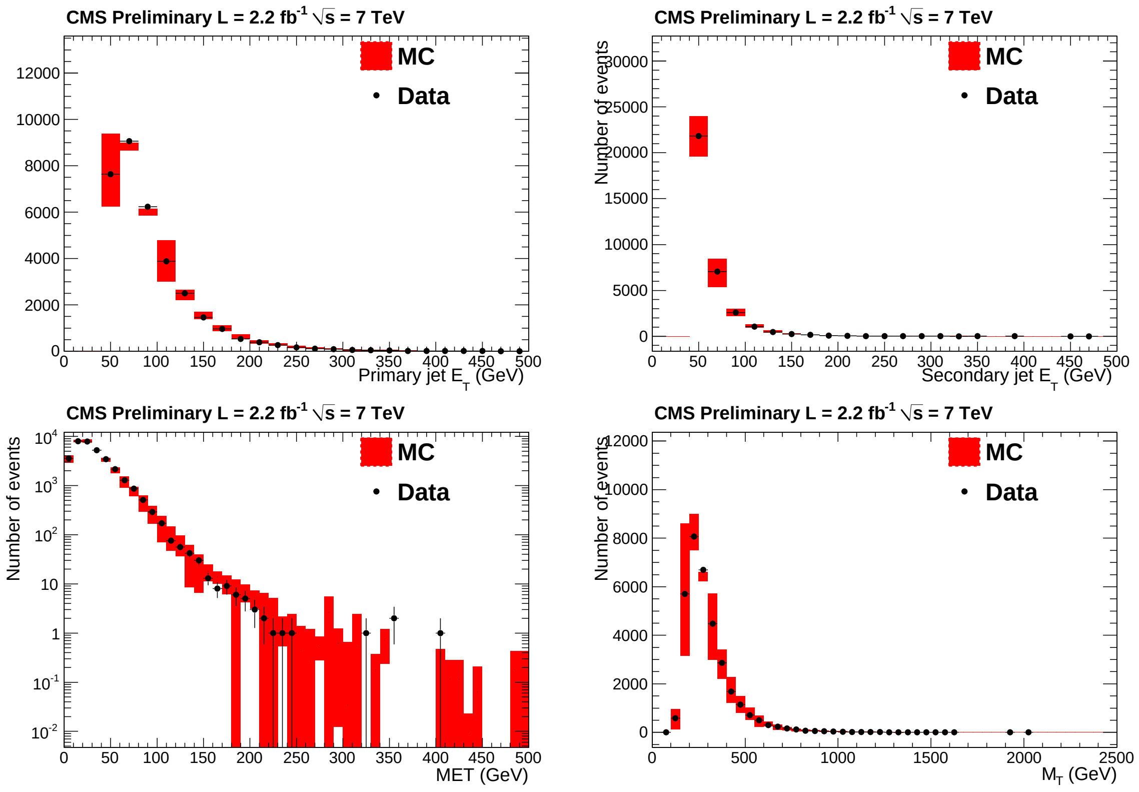

Figure 12: Data (black points) vs MC (red lines) comparisons of all seven ANN input variables and for the events in the CR. Top left : Primary jet p_T (GeV), Top right : Secondary jet p_T (GeV). Bottom left : MET (GeV), Bottom right : Invariant Transverse Mass (GeV).

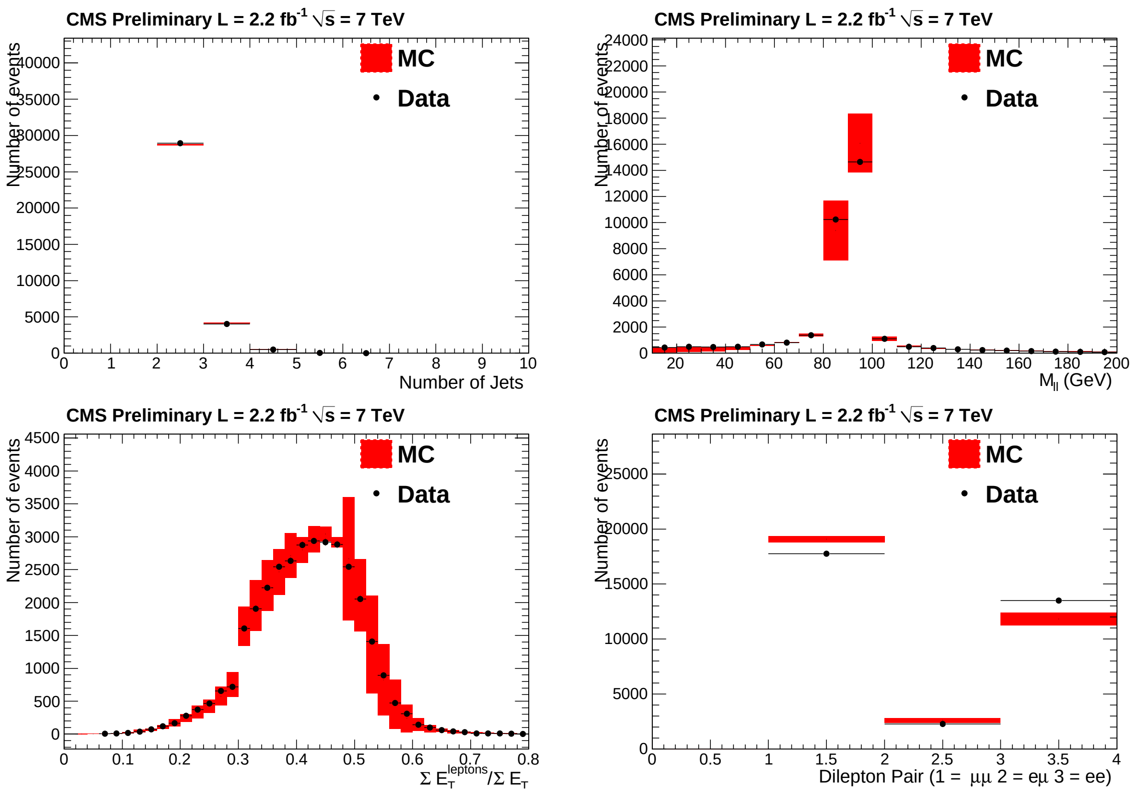

Figure 13: Data (black points) vs MC (red lines) comparisons of all seven ANN input variables and for the events in the CR. Top left : Number of Jets, Top right: Dilepton invariant mass (GeV). Bottom left : Fraction of transverse energy carried by the dilepton system, Bottom right : Dilepton pair type 1 = $\mu\mu$, 2= $e\mu$, 3 = ee.

Data vs Simulated event comparisons of ANN input variables in the signal region : In Figures 14-15 the comparisons of the ANN input variables between the data and simulated events in the signal region are shown. Statistical and systematics uncertainties are included. Simulated events are normalized to the total number of the data events. No pathologies are observed.

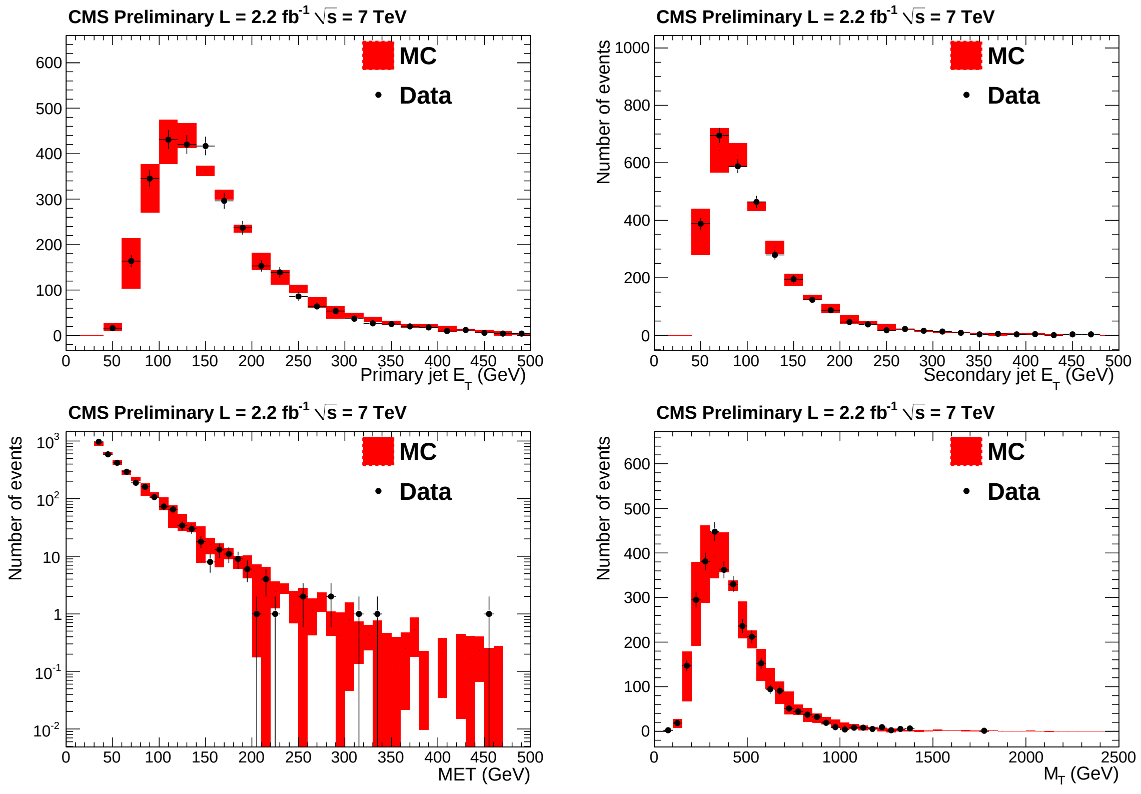

Figure 14: Data (black points) vs MC (red lines) comparisons of all seven ANN input variables and for the events in the SR. Top left : Primary jet p_T (GeV), Top right : Secondary jet p_T (GeV). Bottom left : MET (GeV), Bottom right : Invariant Transverse Mass (GeV).

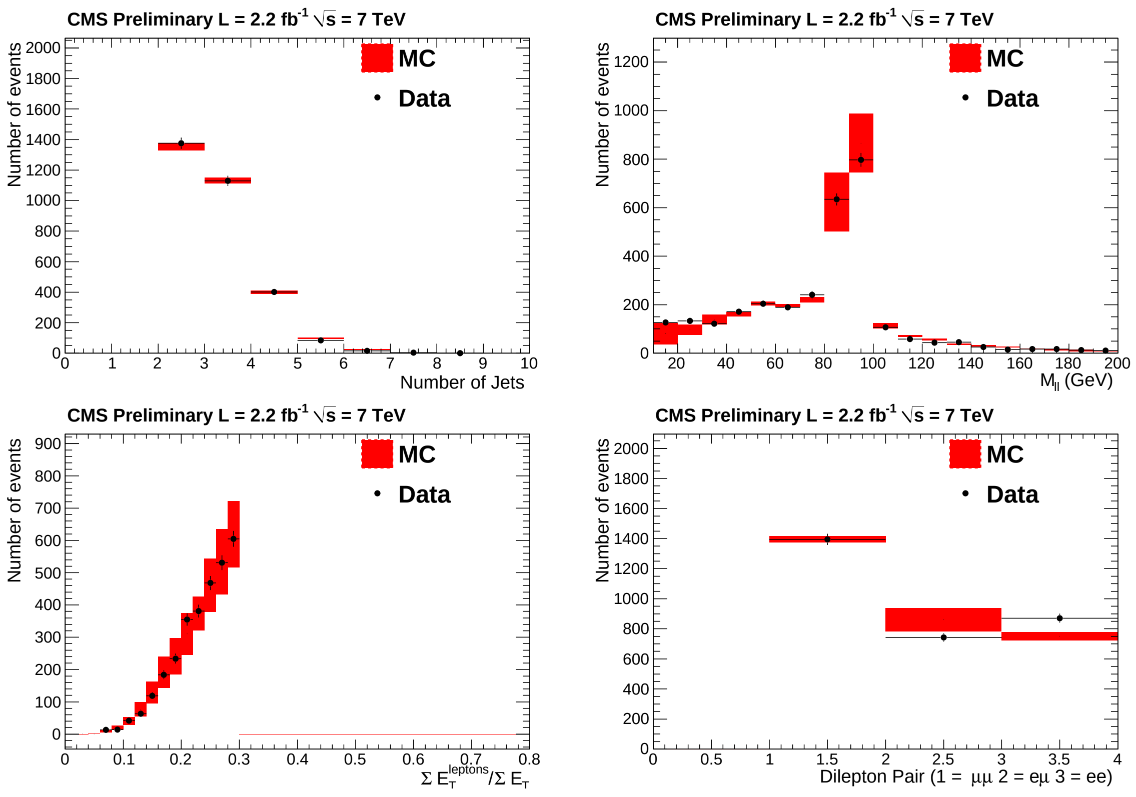

Figure 15: Data (black points) vs MC (red lines) comparisons of all seven ANN input variables and for the events in the SR. Top left : Number of Jets, Top right: Dilepton invariant mass (GeV). Bottom left : Fraction of transverse energy carried by the dilepton system, Bottom right : Dilepton pair type $1 = \mu\mu$, $2 = e\mu$, $3 = ee$.

11 Appendix C

ANN output function :

```
/*
#VAR -*-*-*-*-*-*-*-*-*-*-*-* variables *-*-*-*-*-*-*-*-*-*-*-*-

NVar 7
jet2_et
jet1_et
(sumet_lep/SumEt)
CaloMETPT
cNjets
mass_leplep
MT
============================================================================ */

#include <vector>
#include <cmath>
#include <string>
#include <iostream>

#ifndef IClassifierReader__def
#define IClassifierReader__def

class IClassifierReader {

 public:

   // constructor
   IClassifierReader() : fStatusIsClean( true ) {}
   virtual ~IClassifierReader() {}

   // return classifier response
   virtual double GetMvaValue( const std::vector<double>& inputValues ) const = 0;

   // returns classifier status
   Bool_t IsStatusClean() const { return fStatusIsClean; }

 protected:

   Bool_t fStatusIsClean;
};

#endif

class ReadMLP : public IClassifierReader {

 public:

   // constructor
   ReadMLP( std::vector<std::string>& theInputVars )
      : IClassifierReader(),
        fClassName( "ReadMLP" ),
        fNvars( 7 ),
        fIsNormalised( false )
   {
      // the training input variables
      const char* inputVars[] = { "jet2_et", "jet1_et", "(sumet_lep/SumEt)", "CaloMETPT", "cNjets", "mass_leplep", "MT" };

      // sanity checks
      if (theInputVars.size() <= 0) {
         std::cout << "Problem in class \"" << fClassName << "\": empty input vector" << std::endl;
         fStatusIsClean = false;
      }

      if (theInputVars.size() != fNvars) {
         std::cout << "Problem in class \"" << fClassName << "\": mismatch in number of input values: "
                   << theInputVars.size() << " != " << fNvars << std::endl;
         fStatusIsClean = false;
      }

      // validate input variables
      for (size_t ivar = 0; ivar < theInputVars.size(); ivar++) {
         if (theInputVars[ivar] != inputVars[ivar]) {
            std::cout << "Problem in class \"" << fClassName << "\": mismatch in input variable names" << std::endl
                      << " for variable [" << ivar << "]: " << theInputVars[ivar].c_str() << " != " << inputVars[ivar] << std::endl;
            fStatusIsClean = false;
         }
      }

      // initialize min and max vectors (for normalisation)
      fVmin[0] = -1;
      fVmax[0] = 0.999999940395355;
      fVmin[1] = -1;
      fVmax[1] = 1;
      fVmin[2] = -1;
      fVmax[2] = 1;
      fVmin[3] = -1;
      fVmax[3] = 1;
      fVmin[4] = -1;
      fVmax[4] = 1;
      fVmin[5] = -1;
      fVmax[5] = 1.00000011920929;
      fVmin[6] = -1;
      fVmax[6] = 1;

      // initialize input variable types
      fType[0] = 'D';
      fType[1] = 'D';
      fType[2] = 'D';
      fType[3] = 'D';
      fType[4] = 'D';
      fType[5] = 'D';
      fType[6] = 'D';

      // initialize constants
      Initialize();

      // initialize transformation
      InitTransform();
   }

   // destructor
   virtual ~ReadMLP() {
      Clear(); // method-specific
   }

   // the classifier response
   // "inputValues" is a vector of input values in the same order as the
   // variables given to the constructor
   double GetMvaValue( const std::vector<double>& inputValues ) const;

 private:

   // method-specific destructor
   void Clear();

   // input variable transformation

   double fMin_1[3][7];
   double fMax_1[3][7];
   void InitTransform_1();
   void Transform_1( std::vector<double> & iv, int sigOrBgd ) const;
   void InitTransform();
   void Transform( std::vector<double> & iv, int sigOrBgd ) const;

   // common member variables
   const char* fClassName;

   const size_t fNvars;
   size_t GetNvar()           const { return fNvars; }
   char   GetType( int ivar ) const { return fType[ivar]; }

   // normalisation of input variables
   const Bool_t fIsNormalised;
   Bool_t IsNormalised() const { return fIsNormalised; }
   double fVmin[7];
   double fVmax[7];
   double NormVariable( double x, double xmin, double xmax ) const {
      // normalise to output range: [-1, 1]
      return 2*(x - xmin)/(xmax - xmin) - 1.0;
   }

   // type of input variable: 'F' or 'I'
   char   fType[7];

   // initialize internal variables
   void Initialize();
   double GetMvaValue__( const std::vector<double>& inputValues ) const;

   // private members (method specific)

   double ActivationFnc(double x) const;

   int fLayers;
   int fLayerSize[4];
   double fWeightMatrix0to1[13][8];   // weight matrix from layer 0 to 1
   double fWeightMatrix1to2[8][13];   // weight matrix from layer 1 to 2
   double fWeightMatrix2to3[1][8];   // weight matrix from layer 2 to 3

   double * fWeights[4];
};

inline void ReadMLP::Initialize()
{
   // build network structure
   fLayers = 4;
   fLayerSize[0] = 8; fWeights[0] = new double[8];
   fLayerSize[1] = 13; fWeights[1] = new double[13];
   fLayerSize[2] = 8; fWeights[2] = new double[8];
   fLayerSize[3] = 1; fWeights[3] = new double[1];
   // weight matrix from layer 0 to 1
   fWeightMatrix0to1[0][0] = 3.29177399882218;
   fWeightMatrix0to1[1][0] = -0.245365609699165;
   fWeightMatrix0to1[2][0] = -1.90986652497691;
   fWeightMatrix0to1[3][0] = 2.17021048269155;
   fWeightMatrix0to1[4][0] = 2.44050380027347;
   fWeightMatrix0to1[5][0] = -2.4984497685281;
```

```
fWeightMatrix0to1[6][0] = -2.51054938801178;
fWeightMatrix0to1[7][0] = 0.380597192071282;
fWeightMatrix0to1[8][0] = -0.363076321730308;
fWeightMatrix0to1[9][0] = -1.10527585645504;
fWeightMatrix0to1[10][0] = -1.47503566028071;
fWeightMatrix0to1[11][0] = -3.94849308431189;
fWeightMatrix0to1[0][1] = 0.393030569210624;
fWeightMatrix0to1[1][1] = 2.90248610236407;
fWeightMatrix0to1[2][1] = 1.47918546989089;
fWeightMatrix0to1[3][1] = 1.31183499026862;
fWeightMatrix0to1[4][1] = -0.329085780734975;
fWeightMatrix0to1[5][1] = 2.60503442953516;
fWeightMatrix0to1[6][1] = 0.751506795083867;
fWeightMatrix0to1[7][1] = -1.98569832597909;
fWeightMatrix0to1[8][1] = -2.49079569466835;
fWeightMatrix0to1[9][1] = 0.573608601513552;
fWeightMatrix0to1[10][1] = -2.30105253606289;
fWeightMatrix0to1[11][1] = -1.2260783733781;
fWeightMatrix0to1[0][2] = -1.09190490604195;
fWeightMatrix0to1[1][2] = -2.9033712794073;
fWeightMatrix0to1[2][2] = -4.06246215158626;
fWeightMatrix0to1[3][2] = -0.024398341463822;
fWeightMatrix0to1[4][2] = 2.82043321688032;
fWeightMatrix0to1[5][2] = 1.0540797283071;
fWeightMatrix0to1[6][2] = 0.371572530080945;
fWeightMatrix0to1[7][2] = -1.68327885075904;
fWeightMatrix0to1[8][2] = -1.6303277707654;
fWeightMatrix0to1[9][2] = -2.94588917795664;
fWeightMatrix0to1[10][2] = -0.946941704084185;
fWeightMatrix0to1[11][2] = -0.713428413263343;
fWeightMatrix0to1[0][3] = 1.35992130500694;
fWeightMatrix0to1[1][3] = 0.679007014181195;
fWeightMatrix0to1[2][3] = -1.7953985147428;
fWeightMatrix0to1[3][3] = 6.14732785242734;
fWeightMatrix0to1[4][3] = 0.545902070820528;
fWeightMatrix0to1[5][3] = 2.60026955342781;
fWeightMatrix0to1[6][3] = 0.242854195929274;
fWeightMatrix0to1[7][3] = -0.799971200382343;
fWeightMatrix0to1[8][3] = -1.00233992911731;
fWeightMatrix0to1[9][3] = 1.2133576656284;
fWeightMatrix0to1[10][3] = -0.737803062849777;
fWeightMatrix0to1[11][3] = 1.75986436854496;
fWeightMatrix0to1[0][4] = -1.53478921028282;
fWeightMatrix0to1[1][4] = -0.480026875078978;
fWeightMatrix0to1[2][4] = 1.26838042679403;
fWeightMatrix0to1[3][4] = 1.93795328851792;
fWeightMatrix0to1[4][4] = 2.85833180977158;
fWeightMatrix0to1[5][4] = -0.276474170805862;
fWeightMatrix0to1[6][4] = 0.933706066262378;
fWeightMatrix0to1[7][4] = 3.71445502354918;
fWeightMatrix0to1[8][4] = 0.754969676533977;
fWeightMatrix0to1[9][4] = 0.256773103990782;
fWeightMatrix0to1[10][4] = -2.46883486977909;
fWeightMatrix0to1[11][4] = -1.73993534282581;
fWeightMatrix0to1[0][5] = -0.333538863827411;
fWeightMatrix0to1[1][5] = 1.82145064697017;
fWeightMatrix0to1[2][5] = 2.89344454232447;
fWeightMatrix0to1[3][5] = -2.82510461346494;
fWeightMatrix0to1[4][5] = 0.966521208158374;
fWeightMatrix0to1[5][5] = -1.52963439237833;
fWeightMatrix0to1[6][5] = -1.55067193227353;
fWeightMatrix0to1[7][5] = 0.753144143332884;
fWeightMatrix0to1[8][5] = -1.46708063731015;
fWeightMatrix0to1[9][5] = 0.633458208999106;
fWeightMatrix0to1[10][5] = 4.51743986613654;
fWeightMatrix0to1[11][5] = 3.1451026831006;
fWeightMatrix0to1[0][6] = -2.32823327225954;
fWeightMatrix0to1[1][6] = -2.83347005319015;
fWeightMatrix0to1[2][6] = -0.285451895931534;
fWeightMatrix0to1[3][6] = 1.36124601501126;
fWeightMatrix0to1[4][6] = -0.151514287319446;
fWeightMatrix0to1[5][6] = 2.01176321407178;
fWeightMatrix0to1[6][6] = -0.641968911823231;
fWeightMatrix0to1[7][6] = -1.28366919896421;
fWeightMatrix0to1[8][6] = -0.801376886691149;
fWeightMatrix0to1[9][6] = 3.47622438703557;
fWeightMatrix0to1[10][6] = -1.28398868579992;
fWeightMatrix0to1[11][6] = 1.82307629644828;
fWeightMatrix0to1[0][7] = 1.81368187720653;
fWeightMatrix0to1[1][7] = -1.17384929722833;
fWeightMatrix0to1[2][7] = -0.502005559525891;
fWeightMatrix0to1[3][7] = 6.49571042869284;
fWeightMatrix0to1[4][7] = -0.0496633594651782;
fWeightMatrix0to1[5][7] = 0.194276860592616;
fWeightMatrix0to1[6][7] = 2.64986502839053;
fWeightMatrix0to1[7][7] = -0.418782423114577;
fWeightMatrix0to1[8][7] = -1.40801293877951;
fWeightMatrix0to1[9][7] = 3.26272875101289;
fWeightMatrix0to1[10][7] = -2.60303605388398;
fWeightMatrix0to1[11][7] = 0.458975552858688;
// weight matrix from layer 1 to 2
fWeightMatrix1to2[0][0] = -1.29881483730398;
fWeightMatrix1to2[1][0] = -3.33676580551283;
fWeightMatrix1to2[2][0] = 0.693804071425345;
fWeightMatrix1to2[3][0] = -1.79747751008843;
```

```
fWeightMatrix1to2[4][0] = 0.992064774582075;
fWeightMatrix1to2[5][0] = -0.394525814899103;
fWeightMatrix1to2[6][0] = 1.86135187177854;
fWeightMatrix1to2[0][1] = -0.021938333806063;
fWeightMatrix1to2[1][1] = 0.743404847752841;
fWeightMatrix1to2[2][1] = -0.488646169188674;
fWeightMatrix1to2[3][1] = 1.22052125738976;
fWeightMatrix1to2[4][1] = -2.7143892045582;
fWeightMatrix1to2[5][1] = 0.803501676129198;
fWeightMatrix1to2[6][1] = -1.30709206388702;
fWeightMatrix1to2[0][2] = -3.00462397235305;
fWeightMatrix1to2[1][2] = -0.601593274226534;
fWeightMatrix1to2[2][2] = 0.0354555287823965;
fWeightMatrix1to2[3][2] = 1.44115968597597;
fWeightMatrix1to2[4][2] = 1.35642430900783;
fWeightMatrix1to2[5][2] = -1.41819194351702;
fWeightMatrix1to2[6][2] = -1.11199680675758;
fWeightMatrix1to2[0][3] = 3.41734127422267;
fWeightMatrix1to2[1][3] = 0.871052549534217;
fWeightMatrix1to2[2][3] = 5.01122228407157;
fWeightMatrix1to2[3][3] = -2.487366566567;
fWeightMatrix1to2[4][3] = -4.51515265911672;
fWeightMatrix1to2[5][3] = 1.32903016787925;
fWeightMatrix1to2[6][3] = -0.0633471160920286;
fWeightMatrix1to2[0][4] = 0.711258863848139;
fWeightMatrix1to2[1][4] = -0.999512779730261;
fWeightMatrix1to2[2][4] = -1.5476570745643;
fWeightMatrix1to2[3][4] = 2.4495806628519;
fWeightMatrix1to2[4][4] = -0.129458929494108;
fWeightMatrix1to2[5][4] = 0.766527444137211;
fWeightMatrix1to2[6][4] = -1.05749684556702;
fWeightMatrix1to2[0][5] = 0.626204505224997;
fWeightMatrix1to2[1][5] = 1.30765086371382;
fWeightMatrix1to2[2][5] = 0.907074019624411;
fWeightMatrix1to2[3][5] = 1.18570820034202;
fWeightMatrix1to2[4][5] = -2.43428181181909;
fWeightMatrix1to2[5][5] = 3.14474070710698;
fWeightMatrix1to2[6][5] = 1.00041121885284;
fWeightMatrix1to2[0][6] = -0.360883325844199;
fWeightMatrix1to2[1][6] = 0.405168410525592;
fWeightMatrix1to2[2][6] = -0.875806251522382;
fWeightMatrix1to2[3][6] = -1.46702257731654;
fWeightMatrix1to2[4][6] = 1.7727693341659;
fWeightMatrix1to2[5][6] = 0.290951223827034;
fWeightMatrix1to2[6][6] = 1.16263948862155;
fWeightMatrix1to2[0][7] = 2.35161184442327;
fWeightMatrix1to2[1][7] = 0.423760373224611;
fWeightMatrix1to2[2][7] = -0.480295352879765;
fWeightMatrix1to2[3][7] = 0.949931271448839;
fWeightMatrix1to2[4][7] = 1.99215446893573;
fWeightMatrix1to2[5][7] = 1.85881113723229;
fWeightMatrix1to2[6][7] = -1.57008934961835;
fWeightMatrix1to2[0][8] = 2.07039659572103;
fWeightMatrix1to2[1][8] = 0.303773763459137;
fWeightMatrix1to2[2][8] = -1.02838086803614;
fWeightMatrix1to2[3][8] = -1.15019505545014;
fWeightMatrix1to2[4][8] = -3.94325303988912;
fWeightMatrix1to2[5][8] = -1.76891179710471;
fWeightMatrix1to2[6][8] = -0.112714083435699;
fWeightMatrix1to2[0][9] = 1.48861482817314;
fWeightMatrix1to2[1][9] = -1.39156034637507;
fWeightMatrix1to2[2][9] = 1.39035086798425;
fWeightMatrix1to2[3][9] = 0.320007483525375;
fWeightMatrix1to2[4][9] = -1.74339780204004;
fWeightMatrix1to2[5][9] = 1.66980400983611;
fWeightMatrix1to2[6][9] = -2.81349317384946;
fWeightMatrix1to2[0][10] = -0.318418835815594;
fWeightMatrix1to2[1][10] = 0.285473841942531;
fWeightMatrix1to2[2][10] = -0.79485852294838;
fWeightMatrix1to2[3][10] = 1.09730565320922;
fWeightMatrix1to2[4][10] = -0.869041327837684;
fWeightMatrix1to2[5][10] = 2.1583602437071;
fWeightMatrix1to2[6][10] = 2.2874376091498;
fWeightMatrix1to2[0][11] = -3.09224991804797;
fWeightMatrix1to2[1][11] = 0.126941445542673;
fWeightMatrix1to2[2][11] = 0.799935933607764;
fWeightMatrix1to2[3][11] = 1.588677433542;
fWeightMatrix1to2[4][11] = 2.48296250681092;
fWeightMatrix1to2[5][11] = -0.154657901517641;
fWeightMatrix1to2[6][11] = -1.2772795014775;
fWeightMatrix1to2[0][12] = 0.012977118381234;
fWeightMatrix1to2[1][12] = -0.17845241360871;
fWeightMatrix1to2[2][12] = -1.69955780029875;
fWeightMatrix1to2[3][12] = 0.358046017597222;
fWeightMatrix1to2[4][12] = 1.51641959929158;
fWeightMatrix1to2[5][12] = -0.733148690723457;
fWeightMatrix1to2[6][12] = -1.92068044597187;
// weight matrix from layer 2 to 3
fWeightMatrix2to3[0][0] = 1.98924830620368;
fWeightMatrix2to3[0][1] = -1.7028953677013;
fWeightMatrix2to3[0][2] = -2.64069686558915;
fWeightMatrix2to3[0][3] = 2.21803520200088;
fWeightMatrix2to3[0][4] = -3.18889698009599;
fWeightMatrix2to3[0][5] = 2.0719072060496;
fWeightMatrix2to3[0][6] = 1.53175931560578;
```

```
   fWeightMatrix2to3[0][7] = -0.103438903491271;
}

inline double ReadMLP::GetMvaValue__( const std::vector<double>& inputValues ) const
{
   if (inputValues.size() != (unsigned int)fLayerSize[0]-1) {
      std::cout << "Input vector needs to be of size " << fLayerSize[0]-1 << std::endl;
      return 0;
   }

   for (int l=0; l<fLayers; l++)
      for (int i=0; i<fLayerSize[l]; i++) fWeights[l][i]=0;

   for (int l=0; l<fLayers-1; l++)
      fWeights[l][fLayerSize[l]-1]=1;

   for (int i=0; i<fLayerSize[0]-1; i++)
      fWeights[0][i]=inputValues[i];

   // layer 0 to 1
   for (int o=0; o<fLayerSize[1]-1; o++) {
      for (int i=0; i<fLayerSize[0]; i++) {
         double inputVal = fWeightMatrix0to1[o][i] * fWeights[0][i];
         fWeights[1][o] += inputVal;
      }
      fWeights[1][o] = ActivationFnc(fWeights[1][o]);
   }
   // layer 1 to 2
   for (int o=0; o<fLayerSize[2]-1; o++) {
      for (int i=0; i<fLayerSize[1]; i++) {
         double inputVal = fWeightMatrix1to2[o][i] * fWeights[1][i];
         fWeights[2][o] += inputVal;
      }
      fWeights[2][o] = ActivationFnc(fWeights[2][o]);
   }
   // layer 2 to 3
   for (int o=0; o<fLayerSize[3]; o++) {
      for (int i=0; i<fLayerSize[2]; i++) {
         double inputVal = fWeightMatrix2to3[o][i] * fWeights[2][i];
         fWeights[3][o] += inputVal;
      }
   }

   return fWeights[3][0];
}

double ReadMLP::ActivationFnc(double x) const {
   // sigmoid
   return 1.0/(1.0+exp(-x));
}

// Clean up
inline void ReadMLP::Clear()
{
   // nothing to clear
}
   inline double ReadMLP::GetMvaValue( const std::vector<double>& inputValues ) const
   {
      // classifier response value
      double retval = 0;

      // classifier response, sanity check first
      if (!IsStatusClean()) {
         std::cout << "Problem in class \"" << fClassName << "\": cannot return classifier response"
                   << " because status is dirty" << std::endl;
         retval = 0;
      }
      else {
         if (IsNormalised()) {
            // normalise variables
            std::vector<double> iV;
            int ivar = 0;
            for (std::vector<double>::const_iterator varIt = inputValues.begin();
                 varIt != inputValues.end(); varIt++, ivar++) {
               iV.push_back(NormVariable( *varIt, fVmin[ivar], fVmax[ivar] ));
            }
            Transform( iV, -1 );
            retval = GetMvaValue__( iV );
         }
         else {
            std::vector<double> iV;
            int ivar = 0;
            for (std::vector<double>::const_iterator varIt = inputValues.begin();
                 varIt != inputValues.end(); varIt++, ivar++) {
               iV.push_back(*varIt);
            }
            Transform( iV, -1 );
            retval = GetMvaValue__( iV );
         }
      }

      return retval;
   }

//_______________________________________________________________________
```

```
inline void ReadMLP::InitTransform_1()
{
   fMin_1[0][0] = 30.2489337921;
   fMax_1[0][0] = 807.876464844;
   fMin_1[1][0] = 30.0433063507;
   fMax_1[1][0] = 807.876464844;
   fMin_1[2][0] = 30.0433063507;
   fMax_1[2][0] = 807.876464844;
   fMin_1[0][1] = 43.8707351685;
   fMax_1[0][1] = 1197.57739258;
   fMin_1[1][1] = 36.0710601807;
   fMax_1[1][1] = 1197.57739258;
   fMin_1[2][1] = 36.0710601807;
   fMax_1[2][1] = 1197.57739258;
   fMin_1[0][2] = 0.0169088207185;
   fMax_1[0][2] = 0.299650639296;
   fMin_1[1][2] = 0.0169088207185;
   fMax_1[1][2] = 0.299917250872;
   fMin_1[2][2] = 0.0169088207185;
   fMax_1[2][2] = 0.299917250872;
   fMin_1[0][3] = 30.0934524536;
   fMax_1[0][3] = 719.838317871;
   fMin_1[1][3] = 30.0305995941;
   fMax_1[1][3] = 719.838317871;
   fMin_1[2][3] = 30.0305995941;
   fMax_1[2][3] = 719.838317871;
   fMin_1[0][4] = 2;
   fMax_1[0][4] = 14;
   fMin_1[1][4] = 2;
   fMax_1[1][4] = 14;
   fMin_1[2][4] = 2;
   fMax_1[2][4] = 14;
   fMin_1[0][5] = 0.305924624205;
   fMax_1[0][5] = 410.979614258;
   fMin_1[1][5] = 0.250672847033;
   fMax_1[1][5] = 410.979614258;
   fMin_1[2][5] = 0.250672847033;
   fMax_1[2][5] = 410.979614258;
   fMin_1[0][6] = 71.7072067261;
   fMax_1[0][6] = 2849.3972168;
   fMin_1[1][6] = 69.3445739746;
   fMax_1[1][6] = 2849.3972168;
   fMin_1[2][6] = 69.3445739746;
   fMax_1[2][6] = 2849.3972168;
}

//_______________________________________________________________________
inline void ReadMLP::Transform_1( std::vector<double>& iv, int cls) const
{
if (cls < 0 || cls > 2) {
   if (2 > 1 ) cls = 2;
   else cls = 2;
}
   for (int ivar=0;ivar<7;ivar++) {
      double offset = fMin_1[cls][ivar];
      double scale  = 1.0/(fMax_1[cls][ivar]-fMin_1[cls][ivar]);
      iv[ivar] = (iv[ivar]-offset)*scale * 2 - 1;
   }
}

//_______________________________________________________________________
inline void ReadMLP::InitTransform()
{
   InitTransform_1();
}

//_______________________________________________________________________
inline void ReadMLP::Transform( std::vector<double>& iv, int sigOrBgd ) const
{
   Transform_1( iv, sigOrBgd );
}
```

References

[1] CMS Collaboration ”Search for Physics Beyond the Standard Model in Opposite-Sign Dilepton Events at CMS”, http://cms.cern.ch:80/iCMS/analysisadmin/get?analysis=SUS-10-007-pas-v3.pdf

[2] CMS Collaboration ”Search for Physics Beyond the Standard Model in Z + jets + EmissT events at the LHC” , http://cms-physics.web.cern.ch/cms-physics/public/SUS-10-010-pas.pdf

[3] CMS Collaboration ”Search for new physics with same-sign isolated dilepton events with jets and missing energy”,http://cms.cern.ch:80/iCMS/analysisadmin/get?analysis=SUS-11-010-pas-v2.pdf

[4] CMS Collaboration ”Search for new physics in events with opposite-sign dileptons and missing transverse energy” ,http://cms.cern.ch:80/iCMS/analysisadmin/get?analysis=SUS-11-011-pas.pdf

[5] G. L. Kane, C. Kolda, L. Roszkowski and J. D.Wells, ”Study of constrained minimal supersymmetry”, Phys. Rev. D 49 (1994) 6173.6210.

[6] CMS Collaboration, ”The CMS experiment at the CERN LHC”, JINST 03 (2008) S08003. doi:10.1088/1748-0221/3/08/S08004.

[7] CMS Collaboration, ”CMS TRIDAS Project Technical Design Report, Volume 1, The Trigger Systems”, CERN/LHCC 2000-38 (2000) CMS TDR 6.1.

[8] CMS Collaboration, ”The CMS High Level Trigger”, Eur. Phys. J. C46 (2006) 605.

[9] CMS Collaboration, ”Performance of muon identification in pp collisions at *sqrts* = 7 TeV”, CMS Physics Analysis Summary CMS-PAS-MUO-10-002 (2010).

[10] CMS Collaboration, ”Electron reconstruction and identification at at *sqrts* = 7 TeV”, CMS Physics Analysis Summary CMS-PAS-EGM-10-004 (2010).

[11] M. Cacciari, G. P. Salam, and G. Soyez, ”The anti-kt jet clustering algorithm”, JHEP 04 (2008) 063. doi:10.1088/1126-6708/2008/04/063.

[12] CMS Collaboration, ”Jet Energy Corrections determination at 7 TeV”, CMS-PAS-JME-10-010 (2010).

[13] T. Sjostrand, S. Mrenna, and P. Z. Skands, ”PYTHIA 6.4 Physics and Manual”, JHEP 05 (2006) 026. doi:10.1088/1126-6708/2006/05/026.

[14] J. Alwall et al., ”MadGraph/MadEvent v4: The NewWeb Generation”, JHEP 09 (2007) 028. doi:10.1088/1126-6708/2007/09/028.

[15] S. Agostinelli et al., ”GEANT4: A simulation toolkit”, Nucl. Instrum. Meth. A506 (2003) 250. doi:10.1016/S0168-9002(03)01368-8.

[16] CMS Collaboration, ”Tracking and Primary Vertex Results in First 7 TeV Collisions”, CMS-PAS TRK 10-005 (2010).

[17] CMS Collaboration, ”Calorimeter Jet Quality Criteria for the First CMS Collision Data”, CMS PAS JME-09-008 (2009).

[18] K. Hornik et al., “Neural Networks”, Vol.2, 359-366 (1989)

[19] D.W.Ruck et al., “IEEE Transactions on Neural Networks”, Vol.1, 296-298 (1990)

[20] P.Wasserman, “Neural computing in theory and practice”, Van Nostrad and. Reinhold(1989)

[21] B.Denby, “Tutorial on neural network applkcations in high energy physics” FERMILAB-CONF-92-121-E, (1992)

[22] TMVA Toolkit for Multivariate Data Analysis with ROOT ’http://tmva.sourceforge.net/’

[23] ROOT an object oriented data analysis framework ’http://root.cern.ch/’

[24] CMS Collaboration, ”Determination of Jet Energy Calibration and Transverse Momentum Resolution in CMS”, CMS-PAS-JME-10-011 (2011).

[25] CMS Collaboration, ”Missing transverse energy performance of the CMS detector”, CMS-PAS-JME-10-009 (2011).

[26] CMS Collaboration, ”Measurement of the differential dijet mass cross-section in proton-proton collisions at $sqrts$ = 7 TeV”, CMS-PAS-QCD-10-025 (2011).

[27] CMS Collaboration, ”Measurement of the $t\bar{t}$ production cross section in the dilepton channel in pp collisions at $sqrts$ = 7 TeV with a luminosity of 1.14 fb^{-1}”, CMS-PAS-TOP-11-005 (2011).

[28] CMS Collaboration, " Measurement of Inclusive W and Z Cross Sections in pp Collisions at *sqrts* = 7 TeV", CMS-PAS-EWK-10-002 (2011).

[29] CMS Collaboration, "Measurement of the InclusiveW and Z Production Cross Sections in pp Collisions at *sqrts* = 7 TeV", CMS-PAS-EWK-10-005 (2010).

[30] B. Knuteson and S. Mrenna, BARD: Interpreting new frontier energy collider physics, arXiv:hep-ph/0602101.

[31] N. Arkani-Hamed et al., MARMOSET: The Path from LHC Data to the New Standard Model via On-Shell Effective Theories, arXiv:hep-ph/0703088.

[32] P. Meade, N. Seiberg, and D. Shih, General Gauge Mediation, Prog. Theor. Phys. Suppl. 177 (2009) 143. doi:10.1143/PTPS.177.143.

[33] M. Buican, P. Meade, N. Seiberg et al., Exploring General Gauge Mediation, JHEP 0903 (2009) 016. doi:10.1088/1126-6708/2009/03/016.

[34] A.L. Read, "Presentation of search results: the CLs technique", J. Phys. G: Nucl. Part. Phys. 28, 2693 (2002)

[35] T. Junk, "Confidence level computation for combining searches with small statistics", Nucl. Instrum. Meth. A 434, 435 (1999)

[36] S. Alekhin et al., The PDF4LHC Working Group Interim Report., (2011) .arXiv:1101.0536v1