A Appendix: Inappropriate Gaussian Priors

The majority of work on VAEs propose to parametrize z — both the prior and approximate posterior (encoder) — as a multivariate Gaussian variable. However, the multivariate Gaussian is a uni-modal distribution and can therefore only represent one mode in latent space. Furthermore, the multivariate Gaussian is perfectly symmetric with a constant kurtosis. These properties are problematic if the latent variables we aim to represent are inherently multi-modal, or if the latent variables follow complex, non-linear probability manifolds (e.g. asymmetric distributions or heavy-tailed distributions). For example. the frequency of topics in news articles could be represented by a continuous probability distribution, where each topic has its own island of probability mass; *sports* and *politics* topics might each be clustered on their own separate island of probability mass with zero or little mass in between them. Due to its uni-modal nature, the Gaussian distribution can never represent such probability distributions. As another example, ambiguity and uncertainty in natural language conversations could similarly be represented by islands of probability mass; given the question *How do I install Ubuntu on my laptop?*, a model might assign positive probability mass to specific, unambiguous entities like *Ubuntu 4.10* and to well-defined procedures like *installation using a DVD*. In particular, certain entities like *Ubuntu 4.10* are now outdated — these entities occur rarely in practice and should be considered rare events. When modeling such complex, multi-modal latent distributions, the mapping from multivariate Gaussian latent variables to outputs — i.e. the conditional distribution $P_\theta(w_n|z)$ — has to be highly non-linear in order to compensate for the simplistic Gaussian distribution and capture the natural latent factors in an intermediate layer of the model. However, it is difficult to learn such non-linear mappings when using the variational bound in eq. (1), as it incurs additional variance from sampling the latent variable z. Consequently, such models are likely to converge on solutions that do not capture salient aspects of the latent variables, which in turn leads to a poor fit of the output distribution.

B Appendix: Piecewise Constant Variable Derivations

To train the model using the re-parametrization trick, we need to generate $z = f(\epsilon)$ where $\epsilon \sim \text{Uniform}(0, 1)$. To do so, we employ inverse transform sampling (Devroye, 1986), which requires finding the inverse of the cumulative distribution function (CDF). We derive the CDF of eq. (2):

$$\phi(z) = \frac{1}{K}\sum_{i=1}^{n} 1\left(\frac{i}{n} \leq z\right) K_i + 1\left(\frac{i-1}{n} \leq z \leq \frac{i}{n}\right) * \left(z - \frac{i-1}{n}\right) a_i. \tag{3}$$

Next, we derive its inverse:

$$\phi^{-1}(\epsilon) = \sum_{i=1}^{n} 1\left(\frac{1}{K}\sum_{j=0}^{i-1} K_j \leq \epsilon \leq \frac{1}{K}\sum_{j=0}^{i} K_j\right) * \left(\frac{i-1}{n} + \frac{K}{a_i}\left(\epsilon - \frac{1}{K}\sum_{j=0}^{i-1} K_j\right)\right) \tag{4}$$

Armed with the inverse CDF, we can now draw a sample z:

$$z = \phi^{-1}(\epsilon), \quad \text{where } \epsilon \sim \text{Uniform}(0, 1). \tag{5}$$

In addition to sampling, we need to compute the Kullback-Leibler (KL) divergence between the prior and approximate posterior distributions of the piecewise constant variables. We assume both the prior and the posterior are piecewise constant distributions. We use the *prior* superscript to denote prior parameters and the *post* superscript to denote posterior parameters (encoder model parameters). The KL divergence between the prior and posterior can be computed using a sum of integrals, where each integral inside the sum corresponds to one constant segment:

$$\begin{aligned}
&\text{KL}\left[Q_\psi(z|w_1,\ldots,w_N)||P_\theta(z)\right] \\
&= \int_0^1 Q_\psi(z|w_1,\ldots,w_N)\log\left(\frac{Q_\psi(z|w_1,\ldots,w_N)}{P_\theta(z)}\right)dz && (6)\\
&= \sum_{i=1}^{n}\int_0^{1/n} \frac{a_i^{\text{post}}}{K^{\text{post}}}\log\left(\frac{a_i^{\text{post}}/K^{\text{post}}}{a_i^{\text{prior}}/K^{\text{prior}}}\right)dz && (7)\\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{a_i^{\text{post}}}{K^{\text{post}}}\log\left(\frac{a_i^{\text{post}}/K^{\text{post}}}{a_i^{\text{prior}}/K^{\text{prior}}}\right) && (8)\\
&= \frac{1}{n}\frac{1}{K^{\text{post}}}\sum_{i=1}^{n} a_i^{\text{post}}\left(\log(a_i^{\text{post}}) - \log(a_i^{\text{prior}})\right) \\
&\quad + \log(K^{\text{prior}}) - \log(K^{\text{post}}) && (9)
\end{aligned}$$

In order to improve training, we further transform the piecewise constant latent variables to lie within the interval $[-1, 1]$ after sampling: $z' = 2z - 1$. This ensures the input to the decoder RNN has mean zero initially.

C Appendix: NVDM Implementation

The complete *NVDM* architecture is defined as:

$$\begin{aligned}
\pi(W) &= f^0(E^0 W + b^0),\\
Enc(W) &= f^1(E^1 \pi(W) + b^1),\\
z_{Gaussian} &= \mu^{\text{post}} + \sqrt{\sigma^{2,\text{post}}} \otimes \epsilon_0,\\
z_{Piecewise} &= \phi^{-1,post}(\epsilon_1),\\
z &= \langle z_{Gaussian}, z_{Piecewise} \rangle,\\
Dec(w, z) &= g(-w^{\mathrm{T}} R z),
\end{aligned}$$

where $\otimes$ is the Hadamard product, $\langle \circ, \circ \rangle$ is an operator that combines the Gaussian and the Piecewise variables and $Dec(w, z)$ is the decoder model.[5] As a result of using the re-parametrization trick and choice of prior, we calculate the latent variable z through the two samples, ϵ_0 and ϵ_1. $f(\circ)$ is a non-linear activation function, which was the parametrized linear rectifier (with a learnable "leak" parameters) for the 20 News-Groups experiments and the softsign function, or $f(v) = v/(1 + |v|)$, for Reuters and CADE. The decoder model $Dec(z)$ outputs a probability distribution over words conditioned on z. In this case, we define $g(\circ)$ as the softmax function (omitting the bias term c for clarity) computed as:

$$Dec(w, z) = P_\theta(w|z) = \frac{\exp\left(-w^{\mathrm{T}} R z\right)}{\sum_{w'} \exp\left(-w^{\mathrm{T}} R z\right)},$$

The decoder's output is used to calculate the first term in the variational lower-bound: $\log P_\theta(W|z)$. The prior and posterior distributions are used to compute the KL term in the variational lower-bound. The lower-bound is:

$$\begin{aligned}
\mathcal{L} =& \mathrm{E}_{Q_\psi(z|W)}\left[\sum_{i=1}^{N} \log P_\theta(w_i|z)\right]\\
&- \mathrm{KL}\left[Q_\psi(z|W)||P_\theta(z)\right],
\end{aligned}$$

where the KL term is the sum of the Gaussian and piecewise KL-divergence measures:

$$\begin{aligned}
&\mathrm{KL}\left[Q(z|W)||P(z)\right]\\
=&\mathrm{KL}_{Gaussian}\left[Q(z|W)||P(z)\right]\\
&+ \mathrm{KL}_{Piecewise}\left[Q(z|W)||P(z)\right].
\end{aligned}$$

[5] Operations include vector concatenation, summation, or averaging.

The KL-terms may be interpreted as regularizers of the parameter updates for the encoder model (Kingma and Welling, 2014). These terms encourage the posterior distributions to be similar to their corresponding prior distributions, by limiting the amount of information the encoder model transmits regarding the output.

D Appendix: VHRED Implementation

As described in the model section, the probability distribution of the generative model factorizes as:

$$\begin{aligned}
&P_\theta(\mathbf{w}_1, \ldots, \mathbf{w}_N)\\
&= \prod_{n=1}^{N} P_\theta(\mathbf{w}_n|\mathbf{w}_{<n}, z_n) P_\theta(z_n|\mathbf{w}_{<n})\\
&= \prod_{n=1}^{N} \prod_{m=1}^{M_n} P_\theta(w_{n,m}|w_{n,<m}, \mathbf{w}_{<n}, z_n) P_\theta(z_n|\mathbf{w}_{<n}),
\end{aligned} \tag{10}$$

where θ are the model parameters. VHRED uses three RNN modules: an *encoder* RNN, a *context* RNN and a *decoder* RNN. First, each utterance is encoded into a vector by the *encoder* RNN:

$$\begin{aligned}
h^{\text{enc}}_{n,0} = \mathbf{0},\ \ h^{\text{enc}}_{n,m} = f^{\text{enc}}_\theta(h^{\text{enc}}_{n,m-1}, w_{n,m})\\
\forall m = 1, \ldots, M_n,
\end{aligned}$$

where f^{enc}_θ is either a GRU or a bidirectional GRU function. The last hidden state of the *encoder* RNN is given as input to the *context* RNN. The *context* RNN uses this state to updates its internal hidden state:

$$h^{\text{con}}_0 = \mathbf{0},\ \ h^{\text{con}}_n = f^{\text{con}}_\theta(h^{\text{con}}_{n-1}, h^{\text{enc}}_{n,M_n}),$$

where f^{con}_θ is a GRU function taking as input two vectors. This state conditions the prior distribution over z_n:

$$P_\theta(z_n \mid \mathbf{w}_{<n}) = f^{\text{prior}}_\theta(z_n; h^{con}_{n-1}), \tag{11}$$

where f^{prior} is a PDF parametrized by both θ and h^{con}_{n-1}. Next, a sample is drawn from this distribution: $z_n \sim P_\theta(z_n|\mathbf{w}_{<n})$. The sample and *context* state are given as input to the *decoder* RNN:

$$\begin{aligned}
h^{\text{dec}}_{n,0} = \mathbf{0},\ h^{\text{dec}}_{n,m} = f^{\text{dec}}_\theta(h^{\text{dec}}_{n,m-1}, h^{\text{con}}_{n-1}, z_n, w_{n,m})\\
\forall m = 1, \ldots, M_n,
\end{aligned}$$

where f^{dec}_θ is the LSTM gating function taking as input four vectors. The output distribution is computed by passing $h^{\text{dec}}_{n,m}$ through an MLP f^{mlp}_θ, an

affine transformation and a softmax function:

$$P_\theta(w_{n,m+1}|w_{n,\leq m}, \mathbf{w}_{<n}, z_n) = \frac{e^{(Ow_{n,m+1})^{\mathrm{T}} f_\theta^{\text{mlp}}(h_{n,m}^{\text{dec}})}}{\sum_{w'} e^{(Ow')^{\mathrm{T}} f_\theta^{\text{mlp}}(h_{n,m}^{\text{dec}})}}, \quad (12)$$

where $O \in \mathbb{R}^{|V| \times d}$ is the word embedding matrix for the output distribution with embedding dimensionality $d \in \mathbb{N}$.

As mentioned in the model section, the approximate posterior is conditioned on the *encoder* RNN state of the next utterance:

$$Q_\psi(z_n \mid \mathbf{w}_{\leq n}) = f_\psi^{\text{post}}(z_n; h_{n-1}^{con}, h_{n,M_n}^{enc}), \quad (13)$$

where f^{post} is a PDF parametrized by ψ and h_{n,M_n}^{enc} (i.e. the future state of the *encoder* RNN after processing $\mathbf{w}_n$).

For the Gaussian latent variables, we use the interpolation gating mechanism described in the main text for the approximate posterior. We experimented with other mechanisms for controlling the gating variables, such as defining α_μ and α_σ to be a linear function of the encoder. However, this did not improve performance in our preliminary experiments.

E Appendix: Training Details

Piecewise Constant Variable Interpolation We conducted initial experiments with the interpolation gating mechanism for the approximate posterior of the piecewise constant latent variables. However, we found that this did not improve performance.

Dialogue Modeling We use the Ubuntu Dialogue Corpus v2.0 extracted January, 2016: http://cs.mcgill.ca/~jpineau/datasets/ubuntu-corpus-1.0/.

For the *HRED* model we found that an additional rectified linear units layer decreased performance on the validation set according to the activity F1 metric. Hence we test *HRED* without the rectified linear units layer. On the other hand, for all *VHRED* models we found that the additional rectified linear units layer improved performance on the validation set. For *P-VHRED*, we found that a final weight of one for the KL divergence terms performed best on the validation set. For *G-VHRED* and *H-VHRED*, reweighing the KL divergence terms with a final value 0.25 performed best on the validation set. We conducted preliminary experiments with $n = 3$ and $n = 5$ pieces, and found that models with $n = 3$ were easier to train. Therefore, we use $n = 3$ pieces for both *P-VHRED* and *H-VHRED*.

For all models, we compute the log-likelihood and variational lower-bound costs starting from the second utterance in each dialogue.

F Appendix: Additional Document Modeling Experiments

Iterative Inference For the document modeling experiments, our results and conclusions depend on how tight the variational lower-bound is. As such, it is in theory possible that some of our models are performing much better than reported by the variational lower-bound on the test set. Therefore, we use a non-parametric iterative inference procedure to tighten the variational lower-bound, which aims to learn a separate approximate posterior for each test example. The iterative inference procedure consists of simple stochastic gradient descent (no more than 100 steps), with a learning rate of 0.1 and the same gradient rescaling used in training. For 20 News-Groups, the iterative inference procedure is stopped on a test example if the bound does not improve over 10 iterations. For Reuters and CADE, the iterative inference procedure is stopped if the bound does not improve over 5 iterations. During iterative inference the parameters of the model, as the well as the generated prior, are all fixed. Only the gradients of the variational lower-bound with respect to generated posterior model parameters (i.e. the mean and variance of the Gaussian variables, and the piecewise components, a_i) are used to update the posterior model for each document (using a freshly drawn sample for each inference iteration step).

Note, this form of inference is expensive and requires additional meta-parameters (e.g. a step-size and an early-stopping criterion). We remark that a simpler, and more accurate, approach to inference might perhaps be to use importance sampling.

The results based on iterative inference are reported in Table 5. As Section 6.1, we find that *H-NVDM* outperforms the *G-NVDM* model. This confirms our previous conclusions.

In our current examples, it appears that the *H-NVDM* with 5 pieces returns more general words. For example, as evidenced in Table 4, in the case of "government", the baseline seems to value the plural form of the word (which is largely based on morphology) while the hybrid model actually

G-NVDM	H-NVDM-3	H-NVDM-5
governments	citizens	arms
citizens	rights	rights
country	governments	federal
threat	civil	country
private	freedom	policy
rights	legitimate	administration
individuals	constitution	protect
military	private	private
freedom	court	citizens
foreign	states	military

Table 4: Word query similarity test on 20 News-Groups: for the query 'government".

pulls out meaningful terms such as "federal", "policy", and "administration".

Approximate Posterior Analysis We present an additional analysis of the approximate posterior on 20 News-Groups, in order to understand what the models are capturing. For a test example, we calculate the squared norm of the gradient of the KL terms w.r.t. the word embedding inputted to the approximate posterior model. The higher the squared norm of the gradients of a word is, the more influence it will have on the posterior approximation (encoder model). For every test example, we count the top 5 words with highest squared gradients separately for the multivariate Gaussian and piecewise constant latent variables.[6]

The results shown in Table 6, illustrate how the piecewise variables capture different aspects of the document data. The Gaussian variables were originally were sensitive to some of the words in the table. However, in the hybrid model, nearly all of the temporal words that the Gaussian variables were once more sensitive to now more strongly affect the piecewise variables, which themselves also capture all of the words that were originally missed This shift in responsibility indicates that the piecewise constant variables are better equipped to handle certain latent factors. This effect appears to be particularly strong in the case of certain nationality-based adjectives (e.g., "american", "israeli", etc.). While the *G-NVDM* could model multi-modality in the data to some degree, this work would be primarily done in the model's decoder. In the *H-NVDM*, the piecewise variables provide an explicit mechanism for capturing modes in the unknown target distribution, so it makes sense that the model would learn to use the piecewise variables instead, thus freeing up the Gaussian variables to capture other aspects of the data, as we found was the case with names (e.g., "jesus", "kent", etc.).

G Appendix: Additional Dialogue Modeling Experiments

Ubuntu Experiments We present test examples — dialogue context and model responses generated using beam search — for the Ubuntu models in Table 7. The examples qualitatively illustrate the differences between models. First, we observe that *HRED* tends to generate highly generic responses compared to all the latent variable models. This supports the quantitative results reported in the main text, and suggests that modeling the latent factors through latent variables is critical for this task. Next, we observe that *H-VHRED* tends to generate relevant entities and commands — such as *mount command*, *xserver-xorg*, *static ip address* and *pulseaudio* in examples 1-4. On the other hand, *G-VHRED* tends to be better at generating appropriate verbs — such as *list*, *install*, *pastebin* and *reboot* in examples 1-3 and example 5. Qualitatively, *P-VHRED* model appears to perform somewhat worse than both *G-VHRED* and *H-VHRED*. This suggests that the Gaussian latent variables are important for the Ubuntu task, and therefore that the best performance may be obtained by combining both Gaussian and piecewise latent variables together in the *H-VHRED* model.

Twitter Experiments We also conducted a dialogue modeling experiment on a Twitter corpus, extracted from based on public Twitter conversations (Ritter et al., 2011). The dataset is split into training, validation, and test sets, containing respectively 749,060, 93,633 and 9,399 dialogues each. On average, each dialogue contains about 6 utterances (dialogue turns) and about 94 words. We pre-processed the tweets using byte-pair encoding (Sennrich et al., 2016) with a vocabulary consisting of 5000 sub-words.

We trained our models with a learning rate of 0.0002 and mini-batches of size 40 or 80.[7] As for the Ubuntu experiments, we used a variant of truncated back-propagation and apply gradient clipping. We experiment with *G-VHRED* and *H-VHRED*. Similar to (Serban et al., 2017b), we use a bidirectional GRU RNN *encoder*, where the forward and backward RNNs each have 1000 hid-

[6] Our approach is equivalent to counting the top 5 words with the highest L2 gradient norms.

[7] We had to vary the mini-batch size to make the training fit on GPU architectures with low memory.

20-NG	Sampled	SGD-Inf
LDA	1058	−−
RSM	953	−−
docNADE	896	−−
SBN	909	−−
fDARN	917	−−
NVDM	836	−−
G-NVDM	651	588
H-NVDM-3	607	546
H-NVDM-5	**566**	**496**

RCV1	Sampled	SGD-Inf
G-NVDM	905	837
H-NVDM-3	865	807
H-NVDM-5	**833**	**781**

CADE	Sampled	SGD-Inf
G-NVDM	339	230
H-NVDM-3	**258**	**193**
H-NVDM-5	294	209

Table 5: Comparative test perplexities on various document datasets (50 latent variables). Note that document probabilities were calculated using 10 samples to estimate the variational lower-bound.

den units. We experiment with *context* RNN encoders with 500 and 1000 hidden units, and find that that 1000 hidden units reach better performance w.r.t. the variational lower-bound on the validation set. The *encoder* and *context* RNNs use layer normalization (Ba et al., 2016). We experiment with *decoder* RNNs with 1000, 2000 and 4000 hidden units (LSTM cells), and find that 2000 hidden units reach better performance. For the *G-VHRED* model, we experiment with latent multivariate Gaussian variables with 100 and 300 dimensions, and find that 100 dimensions reach better performance. For the *H-VHRED* model, we experiment with latent multivariate Gaussian and piecewise constant variables each with 100 and 300 dimensions, and find that 100 dimensions reach better performance. We drop words in the decoder with a fixed drop rate of 25% and multiply the KL terms in the variational lower-bound by a scalar, which starts at zero and linearly increases to 1 over the first 60,000 training batches. Note, unlike the Ubuntu experiments, the final weight of the KL divergence is exactly one (hence the bound is tight).

Our hypothesis is that the piecewise constant latent variables are able to capture multi-modal aspects of the dialogue. Therefore, we evaluate the models by analyzing what information they have learned to represent in the latent variables. For each test dialogue with n utterances, we condition each model on the first $n-1$ utterances and compute the latent posterior distributions using all n utterances. We then compute the gradients of the KL terms of the multivariate Gaussian and piecewise constant latent variables w.r.t. each word in the dialogue. Since the words vectors are discrete, we compute the sum of the squared gradients w.r.t. each word embedding. The higher the sum of the squared gradients of a word is, the more influence it will have on the posterior approximation (encoder model). For every test dialogue, we count the top 5 words with highest squared gradients separately for the multivariate Gaussian and piecewise constant latent variables.[8]

The results are shown in Table 8. The piecewise constant latent variables clearly capture different aspects of the dialogue compared to the Gaussian latent variables. The piecewise constant variable approximate posterior encodes words related to time (e.g. weekdays and times of day) and events (e.g. parties, concerts, Easter). On the other hand, the Gaussian variable approximate posterior encodes words related to sentiment (e.g. laughter and appreciation) and acronyms, punctuation marks and emoticons (i.e. smilies). We also conduct a similar analysis on the document models evaluated in Sub-section 6.1, the results of which may be found in the Appendix.

[8] Our approach is equivalent to counting the top 5 words with the highest L2 gradient norms. We also did some experiments using L1 gradient norms, which showed similar patterns.

Word Time-related	G-NVDM G-KL	H-NVDM-5 G-KL	 P-KL
months	23	33	**40**
day	28	32	**35**
time	**55**	22	**40**
century	**28**	13	**19**
past	**30**	18	**28**
days	**37**	14	**19**
ahead	**33**	20	**33**
years	**44**	16	**38**
today	46	27	**71**
back	31	30	**47**
future	**20**	15	**20**
order	**42**	14	**26**
minute	15	34	**40**
began	**16**	5	**13**
night	**49**	12	**18**
hour	**18**	**17**	16
early	42	42	**69**
yesterday	25	26	**36**
year	**60**	17	**21**
week	28	54	**58**
hours	20	26	**31**
minutes	**40**	34	**38**
months	23	33	**40**
history	**32**	18	**28**
late	41	**45**	31
moment	**23**	**17**	16
season	**45**	29	**37**
summer	29	28	**31**
start	30	14	**38**
continue	21	32	**34**
happened	22	27	**35**

Word Names	G-NVDM G-KL	H-NVDM-5 G-KL	 P-KL
henry	33	**47**	39
tim	**32**	27	11
mary	26	**51**	30
james	40	**72**	30
jesus	28	**87**	39
george	26	**56**	29
keith	65	**94**	61
kent	51	**56**	15
chris	38	**55**	28
thomas	19	**35**	19
hitler	10	**14**	9
paul	25	**52**	18
mike	38	**76**	40
bush	**21**	20	14

Adjectives	G-KL	G-KL	P-KL
american	**50**	12	**40**
german	**25**	21	**22**
european	20	17	**27**
muslim	19	7	**23**
french	11	**17**	**17**
canadian	**18**	10	**16**
japanese	16	9	**24**
jewish	**56**	37	**54**
english	19	16	**26**
islamic	14	18	**28**
israeli	**24**	14	**18**
british	**35**	15	**17**
russian	14	19	**20**

Table 6: Approximate posterior word encodings (20-NG). For P-KL, we bold every case where piecewise variables showed greater word sensitivity than Gaussian variables w/in the same hybrid model.

Dialogue Context (History)	Response
Hi . I am installing ubuntu now in my new laptop . In " something else " partitioning , what mount point should I set for a drive which is not root or not home ... → It 's up to you , just choose a directory that will remind you of the contents of that partition . E.G. : if it 's the Windows partition , use /windows . → it 's a new harddrive with full free space . I bought it without windows preinstalled . I want to create drives in which I will only store files .. I mean , not root or not home . What mount point do I set for it ? " /mount " is not shown in drop down menu sorry . I mean /mount I mean , in my desktop , extra drives are mounted in /media do you understand my problem ? Sorry , English is not my native language . → I do :) Just use a directory with a meaningful name and preferably in the root , for example " /files ". Choose something like " /files ". → / for root , /home for home , but what mount point for a file storage partition . wait , let me see if there is a " /files " , thank you .	**HRED:** No problem . **G-VHRED:** It should be in the list of the drive . If you want to be able to mount it . **P-VHRED:** If you want to mount it , you can use the mount command . **H-VHRED:** As far as I know , you can use the " mount " command to mount the partitions .
I have a problem installing Ubuntu 12.04 desktop . Tried desktop with USB -¿ hangs . alternate with usb -¿ hangs . alternate with cd -¿ hangs (even faster) . I tried using different usb drives . Could someonle please help me ? → how are you making the usb ? and whats the pc specs . → I used Universal usb installer → what exactly does it do when it ' hangs ' → Specs : CPU core i7 @ 3.4 Ghz , Sapphire Radeon HD 7800 , 8GB RAM → ATi video ? tried the NOMODESET option yet ? → nothing at all , I also can't witch to console using Ctrl + alt + F not yet Will try that , thanks	**HRED:** you could try the alternate install cd . **G-VHRED:** it might be a problem with your graphics card . You could also try the alternate install cd and see if it works . **P-VHRED:** try ' nomodeset ' or ' nomodeset ' **H-VHRED:** only thing you can do is run the command ' sudo dpkg-reconfigure xserver-xorg ' and see if you get any errors .
anyone know how to add a route in ubuntu ? getting an error on adding a route , and I'm not sure why → pastebin your command and pronlem → <url >	**HRED:** can you pastebin the output of " route -n " please **G-VHRED:** if you want to use pastebin to pastebin the output of " sudo route -n " please pastebin the output of " route " **P-VHRED:** what is the output of : route -n **H-VHRED:** only if you have a static ip address , you need to specify the ip address of the gateway
Any idea why empathy 's not playing notification sounds ? Even though I have ' em ticked in preferences → restarted it yet ? → yar → check pulse to see if the application is muted for some reason ? well Sound settings . → Had sound effects turned off in sound settings , didn't realize that controlled other applications → Ah yea , ive done it a few time it 's annoying My favorite though is recently pulse has been freezing on my desktop and audio will just not be adjustable for like ... 30 seconds or so	**HRED:** I don't think it 's a problem with pulseaudio , but I'm not sure how to fix it **G-VHRED:** Thank you for your help . I will try that . Thanks for the suggestion . **P-VHRED:** I think it 's a bug with pulseaudio , but I don't know how to fix it . **H-VHRED:** as far as I can tell , I have a problem with pulseaudio , but I don't know how to fix it .
Question about setting up drives : Running 12.04 LTS . I have two drives . An SSD that holds the OS for booting , and a second drive " data " for all the data , home directories , etc . The data drive is encrypted with LUKS . With my current setup , whenever I reboot , I get a password entry screen to decrypt the data drive , where my home directory is . This works fine . The problem is that if I reboot remotely , I'm stuck as I can't enter password . If I keep my home directory on the SSD , there won't be enough room . If I use a symbolic link , and the encrypted drive isn't mounted , that creates errors and issues . Suggestions ?? → yes , you can redirect the → Message was cut off ...	**HRED:** if you want to see the data on the drive , you can mount it in /etc/fstab **G-VHRED:** it should be a good place to do that , but if you want to do a clean install , you will need to reboot **P-VHRED:** it 's not a problem , it 's just a bug **H-VHRED:** if you want to see the data on the drive , you can mount it in /etc/fstab
hey guys , does anyone know why i cannot connect to ssh via my external IP ? (connection refused) but ssh -v localhost works fine ? → Have you forwarded the port on your router to the machine you're trying to access ? → It is a work machine , so am not sure of the rules	**HRED:** You need to forward port 22 to your router to forward port 22 to the server . **G-VHRED:** That 's odd . What are you trying to do ? Can you pastebin the output of " sudo netstat " to " pastebin " please ? **P-VHRED:** Can you pastebin the output of " sudo apt-get install openssh-server "? **H-VHRED:** Even if it 's not working , then you need to set the port forward to your router .

Table 7: Ubuntu model examples. The → token indicates a change of turn.

Word	G-VHRED	H-VHRED		Word	G-VHRED	H-VHRED	
Time-related	**G-KL**	**G-KL**	**P-KL**	**Event-related**	**G-KL**	**G-KL**	**P-KL**
monday	3	5	**10**	school	9	16	**50**
tuesday	2	3	**7**	class	11	16	**27**
wednesday	4	11	**13**	game	20	26	**41**
thursday	2	3	**9**	movie	12	20	**41**
friday	9	18	**26**	club	13	22	**28**
saturday	6	6	**13**	party	8	10	**32**
sunday	2	2	**9**	wedding	7	13	**23**
weekend	8	16	**32**	birthday	12	20	**23**
today	18	28	**56**	easter	15	15	**23**
night	16	31	**68**	concert	7	16	**20**
tonight	32	36	**47**	dance	11	12	**21**

Word	G-VHRED	H-VHRED		Word	G-VHRED	H-VHRED	
Sentiment -related	**G-KL**	**G-KL**	**P-KL**	**Acronyms, Punctuation Marks & Emoticons**	**G-KL**	**G-KL**	**P-KL**
good	**72**	**73**	44	lol	**394**	**358**	312
love	**102**	**101**	38	omg	**52**	**45**	19
awesome	**26**	**44**	39	.	386	558	**1009**
cool	14	28	**29**	!	**648**	**951**	525
haha	**132**	**101**	75	?	**507**	**851**	221
hahaha	**60**	**48**	24	*	**108**	**54**	19
amazing	**14**	**38**	33	xd	**28**	**42**	26
thank	**137**	**153**	29	♡	**56**	**42**	24

Table 8: Approximate posterior word encoding on Twitter. The numbers are computed by counting the number of times each word is among the 5 words with the largest sum of squared gradients of the Gaussian KL divergence (G-KL) and piecewise constant KL divergence (P-KL)